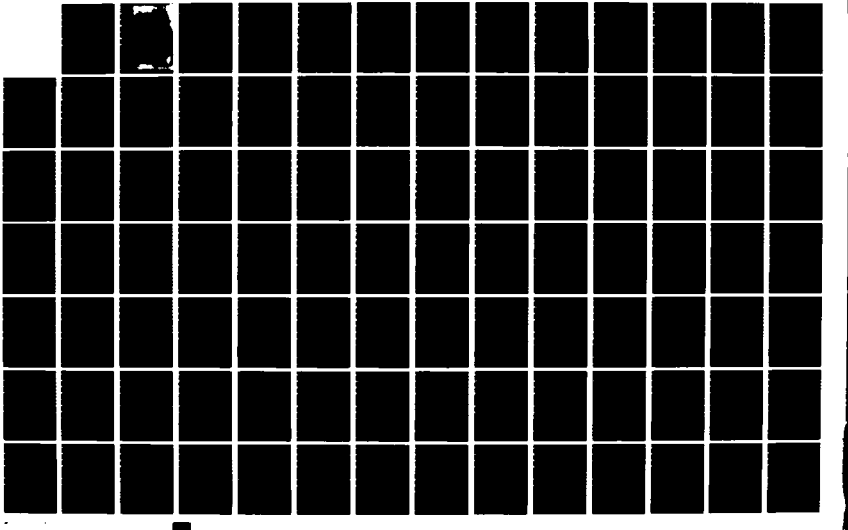
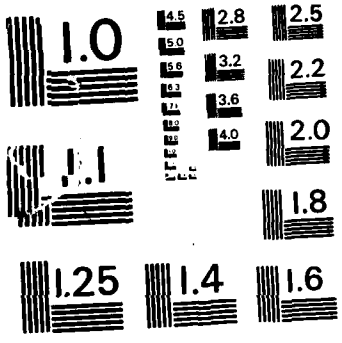
AD-A167 619 DESIGN AND ANALYSIS FOR AREA ROUTING IN LARGE NETWORKS 1/2
(U) SPARTA INC MCLEAN VA 22 APR 86 URW-6-DCA-052
DCA100-84-C-0085

UNCLASSIFIED F/G 9/3 NL

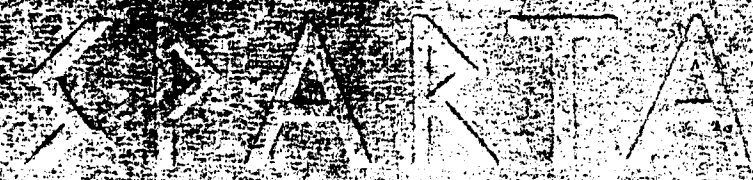




MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

AD-A167 619

Design and Analysis for Area Routing In Large Networks



DTIC FILE COPY

DTIC
ELECTE
MAY 09 1986
D
E



2

Design and Analysis for Area Routing In Large Networks

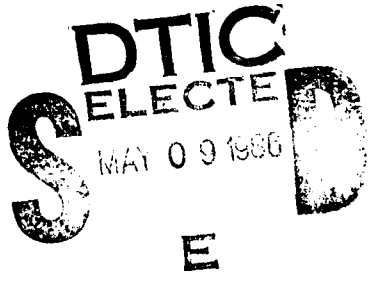
Prepared By

SPARTA, Inc.
7926 Jones Branch Drive
Suite 1070
McLean, VA 22102

April 22, 1986

Prepared For

Defense Communications Engineering Center
1860 Wiehle Avenue
Reston, VA 22090



This document has been approved for public release and sale; its distribution is unlimited.

TABLE OF CONTENTS

1.	INTRODUCTION	1
1.1	Problem Definition	2
1.2	Need for Area Routing in Large Networks	4
1.3	Document Organization	7
2.	ROUTING ALGORITHM DESIGN ISSUES	8
2.1	General Routing Issues	8
2.1.1	The Routing Function	8
2.1.2	Objectives of Routing Algorithms	9
2.1.3	Operational Goals	10
2.1.4	Design Options for Routing Algorithms	14
2.1.4.1	Adaptive and Nonadaptive Routing	14
2.1.4.2	Locus of Control	16
2.1.4.3	Path Selection Mechanisms	17
2.1.4.4	Addressing Schemes	18
2.1.4.5	Routing Information	18
2.1.4.6	Nodal Organization	21
2.2	Design Issues for Area Routing Algorithms	22
2.2.1	Packet Radio Work in Area Routing	22
2.2.2	Specialized Node Functions	24
2.2.3	Integration of Routing Information	26
2.2.4	Clustering Methods for Network Organization	27
2.2.4.1	Recent Research	29
2.2.4.2	Proposal for New Clustering Principle	31
2.3	Conclusions on Algorithm Design Issues	33
3.	DESIGN FOR AN AREA ROUTING ALGORITHM FOR THE DEFENSE DATA NETWORK	35
3.1	Overview	35
3.2	Node Roles	35
3.2.1	Internal Nodes	36
3.2.2	Border Nodes	36
3.2.3	Node Directory	37
3.3	Node Routing Procedures	38
3.3.1	Internal Node Routing Procedures	38
3.3.2	Border Node Routing Procedures	39
3.3.3	Packet Processing Scenarios	40
3.3.3.1	Packet Origination from an Internal Node	40
3.3.3.2	Packet Origination from a Border Node	40
3.3.3.3	Packet Continuation at an Internal Node	40
3.3.3.4	Packet Continuation at a Border Node	41
3.4	Addressing	41
3.5	Update Messages	43
3.6	Area Formation Techniques	45
3.7	Non-Use of Global Routing Nodes	46
3.8	Update Exchange Protocols	46
3.9	Response to Network Partitions	47
3.10	Examples of Area Routing	48
3.11	Estimation of Control Traffic	49
3.12	Design Drivers and Alternatives	52

3.12.1	Design Drivers	52
3.12.2	Design Alternatives	54
3.12.2.1	Border Node Sharing Multiple Areas	54
3.12.2.2	Internal to Border Node Relation	55
4.	SIMULATION AND ANALYTICAL RESULTS	56
4.1	Method	56
4.2	Results	57
4.2.1	Delays Under Flat Routing	57
4.2.2	Delays Under Area Routing	58
4.2.3	Direct Comparison of Flat and Area Routing	61
4.3	Discussion	62
4.3.1	Flat Versus Area Routing	62
4.3.2	Implications for Area Design	63
4.3.3	General Conclusions	64
5.	CONCLUSIONS	66
6.	REFERENCES	67

APPENDIX: GUIDED ADAPTIVE ROUTING FOR THE DDN

A.	GUIDED ADAPTIVE ROUTING FOR THE DDN	1
A.1	Multipath Objectives	2
A.2	Algorithm Description	3
A.3	Avoiding Routing Loops	7
A.4	Packet Switching Simulation	9
A.4.1	Routing Experiment	10
A.4.2	Interpretation of Results	33
A.5	Areas for Further Research	34
B.	PROOF THAT GUIDE ADAPTIVE ALGORITHM IS LOOP-FREE	1

Accession For
NTIS GRA&I [X]
DTIC TAB [ ]
Unannounced [ ]
Justification
By
Distribution/
Availability Codes
Dist | Avail and/or Special
A-1
INSPECTED 3

UNCLASSIFIED
SECURITY CLASSIFICATION OF THIS PAGE

ADA 167619

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188
Exp Date: Jun 30, 19

1a. REPORT SECURITY CLASSIFICATION: UNCLASSIFIED	1b. RESTRICTIVE MARKINGS: NONE
2a. SECURITY CLASSIFICATION AUTHORITY	3. DISTRIBUTION/AVAILABILITY OF REPORT: UNLIMITED
2b. DECLASSIFICATION/DOWNGRADING SCHEDULE	
4. PERFORMING ORGANIZATION REPORT NUMBER(S): URI-6-DCA-052	5. MONITORING ORGANIZATION REPORT NUMBER(S)
6a. NAME OF PERFORMING ORGANIZATION: SPARTA, INC. / 6b. OFFICE SYMBOL (If applicable)	7a. NAME OF MONITORING ORGANIZATION: Defense Communications Engineering Center
6c. ADDRESS (City, State, and ZIP Code): 7926 Jones Branch Drive, Suite 1070, McLean, VA 22102	7b. ADDRESS (City, State, and ZIP Code): 1860 Wiehle Ave., Reston, VA 22090
8a. NAME OF FUNDING/SPONSORING ORGANIZATION: DCEC / 8b. OFFICE SYMBOL (If applicable): R640	9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER: DCA100-84-C-0085
8c. ADDRESS (City, State, and ZIP Code): 1860 Wiehle Ave., Reston, VA 22090	10. SOURCE OF FUNDING NUMBERS: PROGRAM ELEMENT NO. / PROJECT NO. / TASK NO. / WORK UNIT ACCESSION NO

11. TITLE (Include Security Classification)
Design and Analysis for Area Routing in Large Networks

12. PERSONAL AUTHOR(S)
SPARTA, INC.

13a. TYPE OF REPORT	13b. TIME COVERED	14. DATE OF REPORT (Year, Month, Day)	15. PAGE COUNT
Final	FROM 10/22/84 TO 4/22/86	860422	< 112 >

16. SUPPLEMENTARY NOTATION

17 COSATI CODES			18. SUBJECT TERMS (Continue on reverse if necessary and identify by block number)
FIELD	GROUP	SUB-GROUP	Computer Networks, Area Routing, Routing Algorithms, Adaptive Routing

19. ABSTRACT (Continue on reverse if necessary and identify by block

This report presents the design and analysis of an area routing algorithm for the Defense Data Network. The design is based upon differentiating the network into multiple connected areas. Within a single area, current "flat" (SPF) routing would be used. For inter area routes, variations on the SPF scheme would be used (as described in the report.) The analysis measures the degree to which non-optimal paths would be developed as a consequence of area routines, restrictions, finding delay increases on the order of 5 - 10% in small (31 mode) network models. The report also examines the utility of guided, adaptive routing for the DDN, by comparative simulations. These show that guided adaptive routing is competitive with flat routing in most network configurations.

20. DISTRIBUTION/AVAILABILITY OF ABSTRACT: ☒ UNCLASSIFIED/UNLIMITED ☐ SAME AS RPT. ☐ DTIC USERS	21. ABSTRACT SECURITY CLASSIFICATION: UNCLASSIFIED
22a. NAME OF RESPONSIBLE INDIVIDUAL: Mr. Ed Cain	22b. TELEPHONE (Include Area Code): (703) 437-2578 / 22c. OFFICE SYMBOL: R640

DD FORM 1473, 84 MAR — 83 APR edition may be used until exhausted. All other editions are obsolete.

SECURITY CLASSIFICATION OF THIS PAGE
UNCLASSIFIED

1. Introduction

This report has been prepared by SPARTA, INC. for the Defense Communication Engineering Center (DCEC) under Contract No. DCA100-84-C-0085, Analysis and Resolution of Packet Switching Issues. The purpose of the report is to present a design for Area Routing for the Defense Data Network (DDN) and to evaluate the design with respect to other network routing techniques.

"Area Routing" denotes methods and techniques for organizing the DDN into connected sets of packet switching nodes so that every one belongs to an area. Within a single area, routing would be performed as it is currently in the DDN, using shortest path algorithms based upon complete network information at each node. For routing across areas, an algorithm for deciding which areas should be traversed would also be used. Consequently, under area routing it is not necessary for each packet switching node to contain a complete database of information about the network links. The reduction of node database volume has been historically a motivation and criterion for assessment of hierarchical network organization techniques. However, it is at best only a secondary motivation for area routing, because of the direct relation between the sizes of node databases and the amount of control traffic necessary to keep them updated.

Area routing schemes are not the only candidates for solutions to problems anticipated in the operation of very large packet switching networks (even though their motivations are clear). Schemes for simply reducing the frequency of update messages and reducing their contents have also been proposed. The concern with these schemes is that, under them, not enough control can be exerted over the flow of network traffic:

suboptimal routes can be generated because delay estimates are less accurate. This report will also examine the potential consequence of employing an alternative scheme (routing algorithm) that attempts to provide good network responsiveness while economizing on the volume of control traffic. This study is reported in the Appendix. The study report improvements in network performance but with comparable and occasionally greater control traffic volumes. The improvement in performance result from the use of load balancing through secondary routes in response to temporary congestion. This approach has potential to improve network performance in any size network and may provide a favorable tradeoff between overall network bandwidth versus generation of control traffic.

This report also examines the following specific issues with regard to the design of an Area Routing algorithm for the DDN:

- o methods for defining areas;
- o methods for area inter-connection;
- o inter-area delay measurement;
- o formats of inter-area routing information;
- o integration of intra- and inter-area routing information;
- o addressing methods under Area Routing;
- o implications for suboptimal routing;
- o implications for network protocols.

In addition, this report will examine the costs and benefits of Area Routing, compared to both routing under a "flat" network organization (i.e., one with only a single level of organization) and under the current Internetworking methods.

1.1 Problem Definition

This report has been developed in response to the following problem statement from the Request for Proposal No. DCA100-84-R-0800.

> It is highly likely that some of the packet-switching networks which are run by the Defense Communications Agency will have significantly more than 1000 packet switches. The performance which will be offered by the current DDN routing algorithm in such large networks can be expected to suffer because the time

> required to distribute routing updates will increase, making the network less adaptable to changing network conditions. Since the algorithm requires that update information from every node be carried on every network trunk, the overhead on network trunks and in packet switches increases as the network grows. Line delay tables and shortest path tables grow with the network, placing more demands on packet switch memory. Area routing is one approach to solving the large network routing problem. In area routing, the network is divided logically into a number of distinct areas. All nodes exchange complete routing information with the nodes in the same area, but they exchange a much smaller amount of routing information with nodes in other areas. The potential impact associated with exchanging less information between two given areas is that traffic travelling between areas may not be routed in an optimal manner.

This problem statement expresses the tradeoff implicit in the choice of a flat versus area routing scheme for the DDN. Flat routing schemes present unwanted growth in control traffic, reducing the available network bandwidth for user applications. On the other hand, area routing may introduce non-optimal (longer than necessary) paths. Non-optimal paths also reduce the network's available bandwidth, because more of the available bandwidth must be used in transporting data.

This report presents the design of an Area Routing algorithm for the DDN for the purpose of limiting the size of packet switch routing tables and of limiting the amount of use of network trunks to carry update messages for these tables. In addition, this report assesses the impact of Area Routing in terms of non-optimal routes that would be used across areas. Finally, this report examines a multiple path routing scheme under a flat network organization as a design alternative to area routing. The design features which recommend it for use with large networks are discussed, and comparisons of performance simulations are presented for flat, multipath routing versus the current ARPANET routing algorithm.

The overall problem definition is the design of network organization and control techniques that are suitable for network control even under growth to one thousand packet switching nodes. Current organization and control techniques imply that control traffic volume and routing table sizes will grow in proportion to the number of network nodes. As a consequence, more link bandwidth must be devoted to keeping these tables updated. Therefore, the potential impact of this growth is significant consumption of network bandwidth to support routing updates. However, schemes for reducing the amount of network control traffic may result in poorer network performance. Serious considerations must be given to the tradeoffs inherent with alternative routing algorithms.

1.2 Need for Area Routing in Large Networks

The current growth rate of demand for DDN network services as well as the overall goals for network survivability dictate that the DDN will grow to a population of on the order of 1000 packet switching nodes during the 1990s. Consequently, there needs to be an assurance that control techniques are available that will perform adequately in the presence of this node population. Although this report describes the design of a scheme for area routing, specifically aimed at combating size-associated problems, it is useful to ask whether anticipated tradeoffs merit the use of area routing, or whether the performance of existing techniques might be adequate.

The current DDN algorithm involves each node having information on delays of traffic-handling network links and using Dijkstra's algorithm to calculate a minimum delay path over the links. Each node furnishes all other nodes with information about its originating links through brief update messages that are periodically transmitted and then flooded

throughout the rest of the network. The following parameters determine the total amount of network traffic used to carry these update messages:

1. the message size: currently each message consists of 136 constant (overhead) bits plus 16 bits per links reported upon;

2. the message frequency: currently measurements of link delay are made over ten second periods, and updates are transmitted only if a significant change is observed. A minimum of 10 seconds elapse between updates. However, the significance threshold is gradually reduced to zero on successive periods, so that after 50 seconds an update message is generated even if no significant change is found in the delay averages. On the average a node generates link update messages every 30 seconds.

3. the flooding discipline: each node transmits updates on all of its links. Each node that receives an update checks whether it has already received this update; if not, it retransmits it to each of its neighbors (on each of its links). To a good approximation, each update messages flows once on each simplex link in the network.

As a consequence of these design factors, the flow of routing update messages on an "average" network link may be calculated:

```
# nodes generating messages  1000 for the future DDN
X size of each message        184 = 136+(3 links/node*16)
X frequency of message       .033 (every 30 seconds average)
-------------------------------------------------------------
Routing update traffic per link = 6133 bits/second
```

This approximation is independent of the link bandwidth; it amounts to 64 percent of 9600 bps, but only 11% of 56,000 bps and only 0.4 percent of 1.544 Mbps. Bandwidth is a critical network resource, and control traffic may or may not represent a significant fraction, depending upon the link speeds in use. For example, the projected control traffic would represent only 0.4% of T1-speed link capacities.

Previous analyses have also focused on other network resources that may be affected by the choice of routing algorithm: processing power and processor memory. Memory conservation in particular has been the focus of designs for hierarchical network organization. As the number of nodes in a network increases, more routing table entries are required, and more

processing steps are required to compute optimal routes. The question of routing table size was previously motivated by higher costs for computer memory and less available memory in packet switching nodes.

SPARTA has examined the issues of dependence of processor and memory requirements upon network size under the current SPF routing scheme. A worst case example of processing requirements, under current routing techniques, is a network of 1024 nodes with an average branching degree of 3; approximately 10% of the processing power of a BBN C/30E would be required for the best and alternate path calculations. Memory requirements for a 1024 node network with average branching of 3 per node would be around 7,168 20 bit words, or about 2.8% of the available memory.

Technology advances now in progress may be expected to answer these specific concerns about resource consumption in large networks. In the case of memory, technology has already provided ample resource availability to nullify a potential problem. In the case of processing power and communications bandwidth, there is reason to believe that greater availabilities of these resources will be a reality very soon. Improvements in processor architecture include parallel processing and increasingly faster logic families. Parallel processing in particular can be a means for reducing the impact of more complex routing calculations; these could be executed on a separate processor in parallel with packet switching tasks. (Rome Air Development Center has chosen this approach for the development of survivable Internet routing.) Improvements in communications technology included the use of fiber optics for very high bandwidth long haul communications.

It is SPARTA's view that the necessity of an area routing scheme is not a foregone conclusion. The problem posed by network growth to 1000

nodes is real, but a variety of technical means, both improved hardware and alternative routing methods, may be available to handle it. This report uses the opportunity provided by the actual design of an area routing algorithm to assess its impact and the impact of other routing algorithms upon network performance, in order to address the potential advantages and disadvantages of area and other routing algorithms.

The conclusions of this study are that 1. area routing does not exact a high cost in terms of sub-optimal routes, but 2. area routing produces only marginal bandwidth savings over flat routing at projected DDN configurations (as described above, and 3. other types of routing algorithm improvements are potentially more effective than area routing in achieving good network performance. An alternative routing technique is discussed in detail in the Appendices.

1.3 Document Organization

Section 2 provides a general discussion of issues in the design of routing algorithms: what choices may be made and why? Section 2 also reviews some recent research in large network control and performance, assessing the present state of knowledge. Section 3 presents the routing algorithm design itself, describing the network organization, types of data exchanged and procedures performed by the nodes and network controller. Section 4 presents an experimental evaluation of the potential for non-optimal routing in a model network. Section 5 presents the conclusions of this study.

2. Routing Algorithm Design Issues

This section provides an overview of the function, the major design issues, and current research concerning routing algorithms for packet switching networks. Section 2.1 covers issues pertinent to routing algorithms in general, while Section 2.2 focuses on topics peculiar to area routing. Some conclusions concerning the problem of routing in large networks are presented in Section 2.3.

2.1 General Routing Issues

2.1.1 The Routing Function

The service supported by packet switching networks is the transfer of messages from network entry point to network exit point. During transfer messages are broken into one or more packets, which traverse the network individually. The routing algorithm of a packet switching network is the procedure which governs a switching node's selection of an outbound communication link for a packet in transit.

"Adaptive" routing algorithms (i.e., those which dynamically respond to changes in network characteristics such as links going down, locally heavy traffic, etc.) must perform three functions [McQ80]:

1) the measurement of pertinent network characteristics;

2) the distribution of collected measurements; and

3) a calculation, based on the measurements.

It should be kept in mind that these functions are subsidiary to the main task of routing. Whatever its complexity, whether it is a flat or an area scheme, the ultimate purpose of any routing algorithm is to direct a switching node's selection of links to use for packet transmission.

2.1.2 Objectives of Routing Algorithms

The challenging aspect of designing routing algorithms arises from the need for them to perform their link selection function in an optimal manner, so as to provide acceptable service efficiently and at acceptable costs. The requirements and constraints of a network's users will determine how 'acceptable' is defined; these requirements should in turn determine the optimization criteria of a routing algorithm. The primary design goal of a routing algorithm is to direct the routing of traffic so that optimization criteria are the met.

In most networks, the criteria are either the minimization of average packet delay or the maximization of message throughput, though procedures to optimize these criteria are typically subject to constraints such as guaranteed response time and "fairness" (i.e., not allowing maximization of throughput to completely deny service to some users). Whether it is better to optimize for throughput or for response is determined by the predominant traffic load of the network. Though a small number of traffic types, such as Remote Job Entry, will not be very stressing in either domain, most categories of traffic will impose performance requirements. Interactive traffic, for example, cannot tolerate delays of much more than a second, even though the overall volume is low. Hence, to support interactive traffic, minimizing average packet delay is the suitable optimization objective. If, however, a network's traffic consists of high-volume exchanges between file systems (e.g., electronic mail, file transfer, etc), then routes should be chosen which optimize effective bandwidth. For many networks, there are requirements for good performance in both regimes. For example, real-time, transaction

oriented systems (e.g., airline reservation systems), speech transmission, and interactive graphics, all require both low delay and high throughput.

It is well known, if counterintuitive, that routing schemes which are optimal for minimal average delay of individual packets are not optimal for maximal throughput. The divergence is most clearly illustrated in the case in which a routing choice exists between satellite channels, which have very high bandwidth but relatively high propagation delays, and terrestrial channels, which have relatively less bandwidth but negligible propagation delays. A similar divergence between routes optimal for throughput and those optimal for delay will occur if one route differs from another by having both higher latency but also higher bandwidth (i.e., one route allows "pipelining" of packets).

Routes may be constructed which are optimal in regard to characteristics other than delay and throughput. For example, if a method of billing is effected, then routes may be chosen so as to minimize average cost. Routes may also be optimized for security, or for reliability.

2.1.3 Operational Goals

Related to the design objectives of a routing algorithm, there are several operational goals - characteristics of a routing algorithm which are instrumental in satisfying user requirements. For example, a routing algorithm must meet constraints in nodal memory and processor demands. More important, since the bandwidth of transmission links is and will be

the the driving hardware constraint of packet switching systems,* minimizing link utilization due to overhead traffic is a critical operational goal for routing algorithms.

The designs of routing algorithms are subject to a variety of flaws which can have disastrous operational effects. An implicit operational goal for algorithm design is to avoid these errors. Among the most catastrophic errors which a design should preclude is "deadlock," in which throughput drops to nil due to competing requirements for network resources. For example, in the "store and forward" deadlock, no node is able to accept traffic, since each node's available memory has been allocated for use by partially reassembled messages or tandem packets. Timeouts for packets, and preallocation of storage, are techniques which may be used to avoid deadlock.

Adaptive routing algorithms may also exhibit oscillation, which in excessive form is known as "thrashing." Oscillation is said to occur when a routing algorithm causes the repeated reconstruction of routes, on the basis of insignificant changes in network conditions. In a network which is thrashing, the control traffic to reconstruct routes dominates resource use.

Some adaptive routing algorithms--most notably, the Gateway-to-Gateway Protocol (GGP)--intentionally cause nodes to engage in a behavior which is colorfully known as "counting to infinity." Routing algorithms

*In [McQ80], McQuillan, et al. report that nodal memory is not a constraint for ARPANET IMPS. Through the early 1990's, the MILNET is expected to grow to about 1000 nodes. For a 1000 node network, with average degree 3, using C/300 IMPs, assuming flat routing, route computation will require about 5% of processor resource, while route tables and data bases will require about 10% of IMP memory; it is not clear that either of these demands would result in long delays or low throughput. Update transmissions will require about 10% of a 56Kb, 60% of a 9.6 Kb line; use of 10% of available bandwidth for update traffic would adversely impact throughput and average delay.

which cause this behavior are those in which nodes exchange path information, rather than reports about link delays or conditions. The potential for this behavior exists if a node receives information that its neighbor can reach a destination, without knowing that it is its neighbor's next hop. If a destination goes down, then the neighboring nodes will exchange routing updates until a computed distance greater than the network's diameter is reached, at which time they will conclude that neither of them is an appropriate next hop to the the deceased node.

Similar to the "count to infinity," a routing anomaly which can result from poor design is "ping-pong." This behavior is the distributed equivalent of an infinite loop. In a network in which nodes are "ping-ponging," a packet sent by one node generates a reply from a second, which in turn causes the first to again transmit the packet, causing the second to generate an identical reply, and so on, ad infinitum. Routing loops, which develop if the next-hop choices for a given destination of a set of nodes generate a cycle, are a particular form of the ping-pong anomaly.

In distributed adaptive networks, the time required for routing updates to propagate through the network entails that at times nodes will have routing tables inconsistent with those of their neighbors. Because of this, it is only with a high cost in message exchange, and slow adaptation, that the possibility of transient routing loops can be avoided. It is far from clear that an algorithm which allows transient loops will be less efficient or more prone to either high delay or low throughput than one which employs the extensive mechanisms necessary to preclude this phenomena. In the design of the current ARPANET algorithm, the decision was made in favor of simplicity: transient loops may form, but they will not last long [McQ80].

Long-term routing loops, however, are debilitating. Not only does a routing loop sever a communication path between two endpoints, it also overwhelms link resources along the loop. Care must be taken in the design of distributed routing algorithms so that long term loops are avoided.

The problem of coping with potential routing loops is especially acute in distributed, multipath routing algorithms, since it is difficult to avoid the possibility of looping if a packet may be switched from primary to secondary paths while en route. Even if such switching is prohibited, routing loops are possible in some multipath schemes which have inconsistent alternate paths.

In single path routing schemes in which complete route trees are computed, there is a consistency in the routing trees generated at different nodes. For example, consider node A, adjacent to node B. A subtree, T, can be extracted from node A's route tree, whose root is the next-hop neighbor, B. The subtree T from A's routing tree will be congruent to the subtree of B's routing tree which contains all the members of T. In other words, all of A's routes through B to various destinations will be identical to B's routes to those same destinations. In multipath schemes, if this consistency is not maintained for secondary routes, then the secondary routing tables of a series of nodes could send packets on a loop.

In summary, the optimization criteria of a routing algorithm will impact its performance, and should therefore be strictly derived from user needs. Routes generated for one criterion, such as minimal packet delay, will differ from routes constructed for another, such as maximal throughput. However, whatever the objectives of a routing algorithm, its performance will be enhanced if the design meets certain operational goals

such as minimizing link utilization by overhead traffic, minimizing nodal memory requirements, and minimizing nodal processor requirements for route calculations. Furthermore, an operational goal for any routing algorithm is to avoid design flaws such as deadlock. Area routing has been proposed as an approach for achieving operational goals in large networks.

2.1.4 Design Options for Routing Algorithms

There are many options for a designer of routing algorithms. Routing algorithms differ from one another in their degree of adaptiveness, their locus of control, their method of path selection, the type and completeness of network information they exchange, the addressing techniques they employ, and the manner in which they disseminate control information. In addition, what is most germane to this report, routing algorithms may be distinguished by the manner in which they organize the nodes of a network: the nodes may all be members of a single, "flat" set, or they may be organized into a hierarchy of sets. This section will examine the above design options in turn.

2.1.4.1 Adaptive and Nonadaptive Routing

Early commercial packet switching networks such as SNA were nonadaptive: response to configuration changes or otherwise altering a path required offline modifications to the routing tables of a network's the switching nodes. Though SNA's responses to topological changes have since been automated, the operational principles of it and other public data networks and network architectures (e.g., TYMNET) dictate that message routes be set up in advance and used to carry all messages for a given communication session. This scheme is easier to implement than adaptive routing, and it is suited well enough for networks in which customers are charged for specific services provided them and in which operating

conditions are expected to be very stable, and network components are not subject to threats or high failure rates. Recent research has focused upon optimizing techniques for defining the routes, given information about the links and external traffic characteristics and demand volumes [GAVI83].

The ARPANET, which began operation in December, 1969, is adaptive: its routing algorithm dynamically generates new routes to adapt to to topological changes and event to temporary traffic congestion. The first routing algorithms for the ARPANET were based upon update messages and routing tables that communicated nodes' estimates of minimum delays to all other network nodes (rather than updates about direct topological status), at relatively high frequencies--2/3 second between update messages. As operational experience accumulated, the benefits and feasibility of less frequent updates was surmised. In 1980, a second generation routing algorithm was installed in the ARPANET, and it is the current basis of routing in the DDN. It is based upon the exchange of direct topological status and uses update periods between 10 and 50 seconds. The routes generated by both the original and the current ARPANET algorithms attempt minimize delay for individual packets. In the ARPANET, changes in link topology are handled by the same mechanisms which cope with changes in network delay to due traffic. This illustrates a design choice for adaptive networks: whether to separate information concerning topology changes from that reporting on network traffic.

A designer's choice between adaptive versus non-adaptive could be driven by considerations of cost to implement versus the actual adaptability needed in the network. However, it is clear that nonadaptive networks could not meet the survivability requirements of the DDN.

2.1.4.2 Locus of Control

Early commercial packet switching networks employed centralized control. In these networks, a single routing center would calculate all route tables for all nodes, and then send this information to the switching nodes. One problem which these systems faced was that a malfunction in the routing center could disable the entire network. In addition, the processing capacity of the route center and the bandwidth of its incident links would limit the overall performance of the network.

To alleviate the problems of reliability and performance which centralized control entails, most contemporary packet switching networks employ distributed control. In a distributed adaptive network, each node cooperates and shares routing information with its fellow nodes, though the routing decisions are made locally. The route calculations themselves may be performed in toto at each node, as with the current ARPANET algorithm. Alternatively, the calculations may be be distributed, with each node performing a separate portion of the route calculation, as in the original ARPANET.

In addition to "central" and "distributed" control, a third form of control is theoretically possible. In this alternative, known as "local" control, nodes make their routing decisions based only on information from data packets flowing through them. The "hot potato" algorithm, in which a packet is always transmitted on the link with the shortest outbound queue, is an example of a routing algorithm for local control. Backward learning, in which data packets record the nodes which they have visited and nodes develop their routing tables based on this information, is another. Furthermore, routing can be dictated by mixed policies with decisions determined partially from a weighted, random selection of a routing rule.

Analytical studies [YUM81a] [YUM81b] have addressed the design and analysis of such simple routing rules, but these strategies have not been tested in actual networks. The inability either to guarantee some level of performance or to adapt to changing network conditions renders isolated control techniques of little use except as topics of academic or historical interest.

2.1.4.3 Path Selection Mechanisms

In routing schemes which compute secondary or alternate paths, traffic may be divided between these paths on a stochastic or deterministic basis. Most fielded routing algorithms are single path, and hence, deterministic algorithms: at any one time, only one path is maintained between a node and a given destination. However, multipath schemes are currently under study: BBN has proposed multipath routing schemes in [ROS80] and [GARD84], and a new multipath algorithm is described in Appendix A of this report.

As mentioned above, multipath algorithms are especially susceptible to routing loops. Avoidance of loops can be accomplished either at path generation, or by use of routing procedures along a path. In the first approach, alternate paths are generated so that no possibility of cycling exists; the cost of this approach is that generating such paths is computationally expensive, and severely constrains the set of alternate paths. In the second approach, steps are taken at each node to insure that a packet is not sent on a looping path; this approach requires extra processing for each packet routed.

As with area routing, use of multiple paths has been investigated as a means of achieving operational efficiency in networks. However, these potential difficulties in loop-free path selection illustrate that there

are difficulties in implementing multipath algorithms. Some design issues of multipath algorithms are discussed in greater detail in Appendix A.

2.1.4.4 Addressing Schemes

The designer of a routing algorithm must also develop its addressing scheme. The addressing scheme may be physical, in which the network topology determines the address of a node, or logical, in which addresses and network configuration are independent. Logical addressing is more flexible, but requires a mechanism for mapping a logical address into a physical location. Historically, physical addressing has been used because of the ease and naturalness of its implementation. However, the growth and dynamism of network configurations and the potential mobility of host subscribers has heightened awareness of the need for logical addressing. The area routing algorithm design presented in Section 4 employs a logical addressing scheme.

2.1.4.5 Routing Information

The Defense Data Network and other packet switching networks are distributed systems. Nodes make decisions based upon information about the rest of the network, received from sources elsewhere in the network. In the ideal but impossible case, each node would have completely accurate, up-to-date information about the delays on all links in the network. As this information changed, the changes would be instantaneously reflected in the tables at each node. This is at least the direction in which protocols for exchange of routing information (databases) aim--the speediest, most reliable updating process, but within reasonable cost tradeoffs. The question of methods for distributed database updating is largely independent from the design routing calculations, as they constitute a service provided <u>for</u> the calculations. Differing levels of this service

can be provided, at differing costs. The more critical the availability of a network is, the more functionality and expense of a database update protocol can be justified.

The network measurements which will be exchanged, their level of detail, and the manner in which they are to be disseminated, and methods for assuring the correctness of such information when received, are other choices facing the designer. For example, the measures may be link delays, as in the ARPANET. In other routing algorithms, nodes make local measurements or determinations of adjacency, but exchange either partial or complete route tables rather than delay measures. The information which is collected may be flooded (i.e., upon receipt, copied and transmitted on every outbound link), sent point to point, or even "piggy backed" on data packets. The control information may be transmitted periodically or in response to significant network events. The current ARPANET algorithm is a compromise between event-driven and periodic updates, as the threshold for transmitting an update message decreases to zero over time.

Jaffe and Moss [JAF82] presented an algorithm in which updates describing increasing delays (i.e., "bad news") are coordinated, rather than independently distributed as with the current DDN flooding techniques. Coordination of these updates is shown to allow freedom from routing loops and improved adaptation times.

Ford Aerospace & Communications Corporation has recently designed a Gateway Database Protocol for maintaining gateways' routing tables. The important features of this protocol include 1. the use of a transport protocol for transmission of the updates (adding assurance of correct transmission of those updates, at the expense of transmission overhead), 2. update ownership (and hence originator) concept, defining the source of an

update message and limiting how such updates can be propagated. 3. specific aging of update messages, 4. a mechanism for repudiation of erroneous update messages by an "owner" of an update item.

One avenue of research in routing algorithms has been to attempt efficiency through the use of simple and fast decision procedures based on incomplete or approximate routing information. Jaffe [SIAM J. Comput. 14(4): 875-999 (November 1985)] has examined the problem of optimal network utilization for multidestination routing under conditions of less-than-complete node information. (This problem is known to be NP hard even with adequate information.) Jaffe shows that optimality cannot be approached at all closely under conditions of having information about a node's immediate links and those of its neighbors.

Hagouel [HAG83] also has the use of uninformed decision procedures for routing, by examining the performance of routing choices by nodes that are members of clusters. The simpler policies included routing to the nearest border node and probabilistic, random routing to one of several border nodes. The effect of these policies was analyzed by calculating path lengths in many randomly constructed networks. Paths obtained by routing to the closest border node were consistently 7% longer than paths chosen under informed hierarchical routing, and 20% longer than optimal (fully informed) paths. Paths under random decision procedures were consistently 30% longer than completely optimal paths.

Area routing is a special case of limited node information; many network nodes have partial information about the network topology and delays. Consequences of area routing's limited nodal information are discussed in detail in Section 5.

2.1.4.6 Nodal Organization

The manner in which a network's nodes are organized is another choice facing an algorithm designer. In the simplest, "flat" organization, every node maintains routing information on every other node in the network, and routes are calculated by viewing each node as a distinct entity. In an hierarchical scheme, nodes are organized into clusters, which may in turn be grouped into super clusters, and so on. Area routing schemes are the simplest nondegenerate type of hierarchical scheme, in that nodes are grouped into clusters, or "areas," but these areas are not encompassed by higher order groups.

In area routing algorithms, nodes in differing clusters maintain reduced information on one another. In some area algorithms, routes are constructed by viewing a remote cluster as though it were a single node. The potential for nonoptimal routes is greater in area routing than in flat routing, though as is shown in Section 5, the differences may be small. The motivation for adopting an area routing scheme is to improve network performance by reducing overhead traffic, thus making more bandwidth available for data traffic. The reduction in overhead traffic is accomplished by eliminating the requirement that all routing information flow to every node. Hence, the key tradeoff in the decision to employ area or hierarchical routing is whether cost of suboptimal routes is less than the reduction in network control traffic. Area or hierarchical schemes are thus similar in goal and costs to flat schemes with low frequency of updates. However, unlike simply reducing the frequency of updates, an area scheme allows the a higher quality of routing information to be maintained about nearby nodes.

2.2 Design Issues for Area Routing Algorithms

There are many design issues unique to area routing schemes. In particular, the characteristics of the areas, the methods and criteria for forming areas, the designation and roles of special purpose nodes, the integration of routing information from remote areas with local information, and the manner in which information from remote areas is distributed, all must be addressed.

Although this report does represent a new synthesis of design features into a specific area routing algorithm, it is not the only such effort in this direction. Bolt, Beranek and Newman, Inc. (BBN) has also been tasked to design an area routing algorithm for the Defense Communications Agency. Even though the SPARTA design and the BBN design were developed independently, review of a draft version of the BBN design revealed a number of similarities with that presented in Section 3. This similarity is not surprising, since both designs are driven by identical requirements and both adopt similar approaches to major design issues.

The major design issues concerning area routing received their first treatment in research in Packet Radio (PR) nets. Recent work in that area is reviewed in the next section, following which more general area routing topics are discussed.

2.2.1 Packet Radio Work in Area Routing

Packet Radio (PR) nets are characterized by limited memory at each repeater/node and by potentially very dynamic link topologies, yet very large number of nodes. PR nets pose both similar and different sets of problems compared to terrestrial networks. Nodes (i.e., repeater transmitters) typically do have severe memory limitations, necessitating hierarchical schemes to avoid recording information about every network

node. The nature of radio transmission results in both richer (higher branching degree) link topology, due to the broadcast nature of the medium, and more variable link topology, as there are more ways that inter-node reception can be interfered with.

Kleinrock and Kamoun [KLE77] studied the organization of large networks into hierarchical cluster structures, with the objective of minimizing the routing table storage requirements in a single node. Under their design, a nodal routing table would contain entries only for its peer level and the level immediately below. Routing information for increasingly distant nodes is derived by passing up and down the hierarchical structure. This study is regarded as definitive in addressing the compression of nodal routing tables. However, it does imply a hierarchical network topology as well. Such a topology presents problems due to its lack of robustness in the face of link failures and other changes.

Garcia-Luna-Aceves and Shacham [SHA85] have also developed organizational concepts for PR nets. They call for the organization of a network into clusters, and they prescribe protocols for the updating of tables at the global and local levels, with the objective of achieving short path lengths and fast adaptations to topological changes.

The design for the military standard operational radio network, the Joint Tactical Information Distribution System (JTIDS) presents an interesting contrast to the above work. JTIDS stations do come in different sizes, but many must be small, portable and low power in order to be man-portable. At the physical level, the transmission medium is shared via both time slots and frequency division. (Frequency division capabilities are limited by spread-spectrum techniques, however.) A

central network controlling station, with VAX (TM) equivalent power, monitors the mutual reachability of all stations and their link qualities (as measured by error counts), and it assigns primary and alternate relay routes to the stations periodically. The relay route assignments are implemented as assignments to listen during one time slot and to transmit during another. Messages for a node are received (but not retransmitted) on yet another time slot. This design is interesting because it uses the power of a central controller to accomplish the technically difficult function of network organization and control.

The PR network efforts have shown specific approaches to reducing need for routing table space in individual nodes while retaining global connectivity. As such, their solutions may be too extreme for use in terrestrial networks, where much more memory and computing power can be placed in individual nodes. Terrestrial nets also do not share the problems of management of the transmission medium found in PR networks.

In terrestrial networks, the establishment of a link between two nodes is a major (administrative) effort, and is mandated by a need for rich network connectivity. Only in rare circumstances will the availability of terrestrial links vary as quickly as availability of PR links. In contrast, PR networks establish an effectively hierarchical link structure around the nodes, despite a potentially richer connectivity, in order to simplify routing tables and decisions.

2.2.2 Specialized Node Functions

In area routing schemes, more detailed routing information is exchanged between nodes of the same area than for nodes of differing areas. For this reason, it is inescapable that "border nodes," which are adjacent to nodes of differing areas, will perform somewhat differently from

"internal nodes," whose neighbors are all in the same area. The special actions to be performed by border nodes may be fairly complex, or rather simple, depending upon the requirements of the routing algorithm.

At minimum, border nodes filter detailed intra-area routing information of one area from the internal nodes of another area. In some area routing schemes, border nodes maintain more routing information than internal nodes. Area routing schemes may also require that border nodes engage in special protocols with other border nodes, and with the internal nodes of their area or areas.

For the DDN, the number of special functions of border nodes should be constrained by the requirement that every node should have the capability of assuming the role of a border node. Furthermore, the border node functions should not require complex processing to the point that the border nodes become bottlenecks.

Many area routing schemes which have been suggested require another type of specialized node, which has been variously called a "Global Routing Node" [SHA85], "Coordinating Node" [HIL81], "Cluster Head" [BAK83], "Station" [KAH78], or "Master Node" [JAF84]. In this discussion, we will refer to such nodes as 'GRNs,' after Shacham. The exact function of a GRN differs from algorithm to algorithm, though in general they provide a degree of centralized control within their areas, and perhaps also are storehouses and distributors of "global," or extra-area, routing information. As with border nodes, area routing schemes for the DDN which include GRNs should keep the functions simple enough to avoid congestion at the GRN, and to avoid the necessity of special purpose hardware for GRNs. Furthermore, if GRNs are required for an area to function, then there must

be means for nodes to detect the failure of their GRN. There must also be protocols for electing an alternative to this role.

2.2.3 Integration of Routing Information

Area routing algorithms must define the procedures by which both local and remote routing information is exchanged. Much as flat routing algorithms differ in what network conditions are measured and reported, area routing schemes will differ from one another in regard to the routing information that is exchanged, as well as in which nodes gather and which nodes receive that information.

The fact that routing information is represented in a less detailed manner for destinations which are outside of a node's area raises another design issue for area routing algorithms: the integration of routing information from the local and area levels. In analyzing area routing algorithms along this line, Callon and Lauer characterize them as either pure or semi hierarchical, depending upon the commonness of route metrics and routing algorithms at the area and node levels [CAL85]. In pure hierarchical schemes, the integration is minimal: a completely separate algorithm is used to construct routes from area to area than is used to generate paths within an area. For routing to destinations which are in remote areas, the first problem solved is the generation of an area to area path. The problem of traversing each intervening area is solved locally, independently of the global algorithm or the other local algorithms. The correlation between routing metrics used at the higher level will be a rough approximation, at best, of local routing metrics. Internet routing of IP datagrams is an example of a pure hierarchical routing approach.

In semi hierarchical routing algorithms there is higher integration of route calculations between the two levels. In such schemes, interior

nodes have access to estimates of trans-area distances for remote areas, which are directly comparable to the measures they have for intra-area distances. In some semi hierarchical schemes (e.g., see Section 4), an interior node can determine the optimal border nodes to use for a particular destination. The major tradeoff between pure and semi hierarchical schemes is similar to that between flat area algorithms: semi hierarchical schemes generate more optimal routes, at the expense of a higher volume of control traffic. However, semi hierarchical algorithms will not equal flat algorithms in path optimality, primarily because of paths generated because routes across an area will be constrained to reside completely in that area. Also, the estimates of trans-area distances employed by semi hierarchical algorithms will tend to be slightly more dated than those of flat algorithms, since in a semi hierarchical algorithm border nodes must both gather and disseminate the information. In flat routing algorithm, a separate process of dissemination *would not be* required.

Semi-hierarchical algorithms can be susceptible to oscillation, due to the fact that the cumulative effect of a a number of small changes in delay can be unexpectedly large. If delay across each of a series of links is stochastic, then the total delay across those links will also be stochastic. For some distributions (e.g., the Gaussian, the Poisson), the variance of a sum of independent random variables is the sum of variances. Hence, semi hierarchical algorithms should be designed to avoid generating trans-area update messages at too high a frequency.

2.2.4 Clustering Methods for Network Organization

In the general case of hierarchical routing, one of the first questions to address is the optimal number of levels of clustering.

Clearly, the answer to this problem depends on the design objectives of the algorithm. Kleinrock and Kamoun report that, for minimizing the size of routing tables, and hence route algorithm demands on nodal memory, the optimal number of clusters is roughly equal to ln(N),* where N is the number of nodes [KLE77]; they suggest that each cluster have either two or three entities (sub clusters or nodes) as members. Intuitively, this level of clustering seems excessive for optimizing for throughput or delay. Research in area routing should incidentally suggest whether network performance gains could be achieved from increased levels of clustering.

In investigating area routing, we have fixed the number of levels at two. However, many other questions which pertain to the desired characteristics of multilevel clusters also pertain to areas. A fundamental question concerns the membership restrictions for areas. In particular, do the areas partition the set of nodes, or will some nodes be members of multiple areas? Another issue concerning cluster characteristics, which is unresolved for most network performance goals, is the optimal number of nodes per area.

Of course, the number of nodes per cluster is not likely to be the primary criterion for optimal clustering, as network topology will no doubt have a large impact on cluster formation. Hilal and Liu suggest that areas should be highly connected [HIL81]; a formalization of this criterion could perhaps be based on minimizing the cardinality of cut sets for each area

---

*The integer solution give by Kleinrock and Kamoun is that the optimal number of clusters will be either ln(N)/ln3, (ln(N/2)/ln3) +1, or (ln(N/4)/ln3) +2; they reach this conclusion by first demonstrating that "there exists an optimum aggregation such that at most two levels have two clusters, while all others have 3," [KLE77, p.163]. So, if m is the number of clusters, and x restrained to be an element of (0, 1, 2), then the optimal m is the least such that $3^{m-x}2^{x} >= N$.

and the remainder of the network. An analysis of techniques for optimal area formation is given in the next section.

Even when criteria for area formation are in hand, the formation of areas poses questions to the designer of an area routing scheme. A significant decision to be made is whether dynamic area formation is a performance requirement, or whether static areas will satisfy user requirements. If a dynamic scheme is needed, then a protocol for cluster formation must be developed.

Even if static areas are used, a fully developed area routing scheme should include mechanisms for partition detection and recovery. Given small enough areas, partitioning could occur as the result of failure of only a handful of nodes or links. In a static scheme, a partial partition recovery might consist of announcing the creation of a new area. An implication of the necessity of coping with partitions is that area membership must not be fixed for a given address (e.g., as in the scheme used by Kleinrock in Kamoun in [KLE77], in which subfields of addresses determined area membership). Instead, some form of dynamic area assignment should be possible.

2.2.4.1 Recent Research

There are a variety of methods for organizing a network into clusters that define its areas or other types of hierarchical groups. The actual structure of the network clustering has its impact both upon the size of routing tables and upon the resulting network performance. Kleinrock and Kamoun [KLE77] examined the potential for sub-optimal path lengths under hierarchical clustering, for example, and found that path lengths under their clustering and routing scheme approached optimality as the number of nodes becomes large. However, the lengths of paths is only

one component in the determination of network performance. A second major component is the utilization of the paths. Clearly, worse overall performance for the same set of paths would be realized if the less optimal paths were used more frequently.

The problem of defining network clusters so as to optimize some measure (e.g., routing table minimization) has been shown to be of NP-hardness [HAG83] [GARE79]. To date the measures to be optimized have been simple and have been directly related to consequent routing table size requirements. The case of determining optimal clusters where network performance characteristics are objectives is harder still and has not been addressed to date.

In either event, the complexity needs to be circumvented through the use of search-pruning or "heuristic" techniques for cluster determination. A simple and expedient solution is "administrative decision" accomplished perhaps by regarding a map of the network topology and encircling groups of nodes in order to form areas, and then assuring their connectivity and potential for routing table size (and consequently routing update traffic) reduction through hand calculations.

On the other hand, Hagouel [HAG83] studied specific algorithms for network cluster determination. The algorithms addressed procedures with lower computational complexity, yet with the potential for design good clusters. (The latter potential was assessed through simulation studies.) The algorithm principles include the following:

1. "growing" clusters/areas around selected origins to guarantee cluster connectivity;

2. selecting as cluster/area centers nodes with the lowest branching degree (fewest numbers of attached links); this assures against later clustering problems by leaving available nodes with high connectivity;

3. governing cluster formation through a maximum cluster size, e.g. 20 nodes.

The clustering schemes were evaluated on the basis of 1. average number of nodes per cluster as a function of the maximum allowed nodes, 2. size of node routing tables that resulted from the cluster structure, and 3. number of Border Nodes that result from the cluster structure.

Hagouel [HAG83] presented an evolution of a clustering algorithm that evolved from experience with performance of the algorithms upon randomly generated network configurations, thereby removing potential bias that would result from examining clustering on only one network structure. The principles gained during observing algorithm performance are embodied in his final algorithm version.

Interestingly, the results of application of his algorithm to a 19-node model of the ARPANET used in simulation studies described in Section 4 suggest that developing clusters from nodes of low degree can foster the creation of too many border nodes. Of the 19 nodes grouped into clusters of maximum size 10, 2 clusters were formed, and 9 of the 19 nodes were border nodes. Nodes of higher degree are incorporated into clusters during final cluster-building phases. These nodes of higher degree are more likely to have neighbors outside the area.

2.2.4.2 Proposal for New Clustering Principle

Although the sensitivity of network performance to network clustering algorithms has yet to be examined, there is yet another clustering principle worthy of incorporation into an algorithm. When a network is organized into clusters, the routing algorithm allows shortest-path routing within individual clusters; shortest paths are only approximated when routing beyond individual clusters is performed. Therefore a cluster structure that attempts to encompass major traffic

flows within the same clusters may offer better network performance on a traffic-weighted basis. That is, more of the offered traffic will be carried on optimal routes under such an organizing principle.

The basis for traffic-oriented cluster organization would be the tabulation of expected communication volumes between network nodes. There are two interpretations of this: 1-- end-to-end traffic volume between the two nodes, and 2-- total traffic volume between the two nodes, including both end-to-end and in-transit traffic that happens to use the two nodes in question. (Determination of the latter requires assumptions or knowledge of existing routing procedures, however,) Given clusters that tend to contain either type of node pairs, greater volumes of traffic would be routed over optimal routes.

The construction of a cluster would proceed by initially sorting a list of node-to-node traffic volumes and choosing the first pair. If the minimum hop distance between this pair is less than the maximum cluster size, then an initial cluster is formed from that path. This cluster would be further augmented by addition of neighbor nodes (of path element nodes), with preference to containing even more traffic within the cluster.

Alternatively, construction of the cluster could be accomplished by choosing multiple node pairs with significant inter-communication volumes and hop-proximity such that an area could be formed from them (within maximum size limits). Such an area would also accomplish routing much traffic over optimal routes. However, its computational complexity would again approach NP hardness [GARE79].

Unfortunately, these schemes may be vulnerable to a particular type of routing non-optimality associated with network clustering. One doctrine of area routing is that whenever a destination lies in the same area, the

message route will be absolutely (rather than preferentially) constructed within that area. Non-optimal paths can arise when there is higher congestion or other form of delay within an area relative to bordering areas, such that an optimal route would leave the source/destination area to traverse a less congested area. The probability of this occurrence would be promoted by the traffic-oriented clustering schemes.

SPARTA recommends further study toward defining principles that are useful for network clustering. To date, there has been no systematic examination of the relation between network performance and clustering principles other than the calculation of structural parameters, such as routing table sizes. The effects compounded by traffic loading have yet to be examined.

2.3 Conclusions on Algorithm Design Issues

This brief examination of design issues and recent research in packet routing provides several insights into the general state of knowledge about problems in large networks.

1. The operational goals of efficient, error free performance by routing algorithm are largely independent of the user type-of-service requirements for a network. Area routing and multipath routing are two distinct approaches to meeting the operational goal of efficient resource utilization.

2. Examination of the design issues shows several obvious choices for area routing for the DDN: the algorithm should be adaptive; the locus of control should be of limited span (i.e., over a single area); and a distinction should be made between the routing information exchanged within and between areas. For survivability and simplicity, an area routing scheme for the DDN should not include GRNs. In order to utilize lower level delay measures from different areas, a semi-hierarchical approach is appropriate.

3. It is interesting to note the extent of the contributions from Packet Radio Network research. These networks tend to be large, they have potentially volatile link topologies, and the resources of individual nodes must be conserved. PR network research has developed techniques to address these problems, but selectivity is required in adapting them to terrestrial networks. In terrestrial nets, link topologies are not as volatile, and node resources are not so limiting.

4. Both the results from PR research and research in general less-informed routing decision techniques have helped to quantify non-optimal routing performance. In general, non-optimal methods show promise of staying reasonably close (within a factor of 2) to optimal ones.

5. The area of update algorithms and protocols appears to be a promising one, given the findings of Jaffe and Moss, but a difficult one to explore, owing to the hurdles of actually implementing designs. Given the importance of this function in assuring consistent node databases, it is worth serious further investigation.

6. The definition of network areas (also referred to as "clustering") remains an unexplored area. Work to date has examined its optimization only with respect to routing table size. This approach does promise better network performance through lower control traffic volume. However, the resulting sensitivity of network performance to cluster definition choices has yet to be examined.

3. Design for an Area Routing Algorithm for the Defense Data Network

3.1 Overview

This section describes a design of a routing algorithm to be used in large networks that are organized into areas consisting of multiple packet switching nodes. The specific design topics addressed include the following:

- o definition of switching node roles, the routing functions performed by the switching nodes according to their roles, techniques for addressing the nodes
- o the messages and protocols for routing information update exchanges among the packet switching nodes, and
- o techniques for area formation and re-formation.

The area organization is a two-level organization; there is no level lower than that of a single packet-switching node and no level above that of an area (a connected collection of packet switching nodes). For both levels of organization, the principles of finding the shortest path between two nodes in a network are employed. Within a single area, the implementation of this principle is identical to that used in "flat" networks. Among the areas that comprise a whole network (every node is a member of one and only one area), the principle is applied over a set of border nodes (as defined below), thereby finding optimal paths over several areas.

3.2 Node Roles

This design for area routing is based upon a need to have strong similarities among the two node roles described below. Both node roles require similar hardware and software support. The roles described below can be performed by particular software modules within a common overall packet switch architecture. In particular, both internal and border nodes

are expected to support several subscriber hosts, packets entering and leaving the network, and to support the relaying of in-transit packets.

3.2.1 Internal Nodes

An internal node is one with no links to a node in another area. A node can detect that it is an internal node based upon the names of nodes in its area and the names of nodes that are its direct neighbors. (This information is downloaded to the node during its initialization.) If all neighbors are also in the same area, then the node "realizes" that it is an internal node.

An internal node sends and receives network topological information (i.e., link existence and delays) to and from other nodes in its area. However, it has no detailed network topological information about nodes within other areas, and it has very limited information about access to other areas. Specific access techniques are described below.

3.2.2 Border Nodes

A border node is one which has at least one link into another area. A node can detect that it is a border node also from the names of nodes in its area and the names of its direct neighbors (provided during the node initialization). If any direct neighbors are not members of the same area, then the node "realizes" it is a border node.

The border nodes are organized into a virtual network for the purpose of calculating minimal delay paths among border nodes. The "virtual network" consists of the set of border nodes, the border node's inter-area links (direct links to a border node in another area) and the border node's intra-area paths to other border nodes in the same area. (These paths are regarded as intra-area "links.") A border node sends and receives among other border nodes information about the "virtual" network.

This information describes delays to neighboring border nodes that are either directly or "virtually" connected. Neighboring border nodes are either directly connected to one another via an inter-area link, or else they are members of the same area. In the latter case, a best path between two border nodes can be calculated across the area.

A border node also reports to the internal nodes within its own area its estimates of delays from itself to remote areas. This information is derived by selecting the border node in the target area which gives the least delay from the source border node among all border nodes in the target area and reporting the delay for the remote border node to the internal nodes.

3.2.3 Node Directory

In order to minimize the configuration differences between border nodes and internal nodes, provisions are made to select routing procedures based upon a node's role. The Node Directory maintains current information about each node's area membership. In order to select its routing procedures, every node must know its own area, what type of node it is and the area of the destination node. This is accomplished using a Node Directory, which is a look-up table configured in each node at start-up in the following manner:

> At start-up a description of the network consisting of the nodes, their links and the clustering of the areas is loaded. After this information is loaded, each node and what area it is in is entered into the Node Directory. The nodes then determine their type of node, internal or border node, in the following manner. Using the information in the Node Directory each node determines if any of its neighbors belong to another area. If so, it sets its "ISBN" flag in the Node Directory.

The node directory contains only node existence information. Information about node connectivity is established and maintained in real time as part of the routing algorithm.

3.3 Node Routing Procedures

3.3.1 Internal Node Routing Procedures

Internal nodes contain database information concerning the distance from it to all other nodes in the same area plus entries for distance to each area (border node entrance into that area with minimal path length) via a particular border node (border node exit from present area). The database is updated using only updates from nodes in the same area. A routing tree is generated from this database using the SPF algorithm. (The SPF algorithm [DIJK59] is a standard method for generating optimal paths between two network nodes, given distance metrics for each link in the network. The choice of such an algorithm is not a design issue here.)

We use time delay estimates for the "distance" concepts. That is, the routing algorithm attempts to minimize the expected delay experienced by packets traveling between network nodes. Delays result from the time necessary to transmit packets over links of finite bandwidth and the time necessary for recently arrived packets (at a node) to await transmission of other packets that had previously arrived (i.e., queuing delays). Although other distance metrics, such as throughput availability, can be used, delay is used here for simplicity.

The area entries are treated as nodes and always appear as leaves in the tree. When an internal node wishes to route to a destination in another area, it consults the routing tree as it would for a destination within the area. An internal node estimates distances to remote areas, based upon update messages received from each border node in the same area, by adding its own distance to a particular border node to that border node's distance-to-area estimate. An internal node updates its distance-to-area

estimate whenever a current estimate is better than the existing one in the database.

3.3.2 Border Node Routing Procedures

A Border node contains two databases used to construct routing trees: one for within-area routing and one for inter-area routing. The within-area database contains information about its distance to every other node in the same area and is maintained using updates from nodes in the same area and the resulting SPF calculations. The inter-area database contains information concerning interarea links (to other border nodes) and delay estimates (over best paths, "virtual links", between border nodes in the same area) for each area and is maintained using Border Node updates. Again, delay estimates are used as the distance measure for actual links and virtual links.

A Border Node generates two separate routing trees: a within-area routing tree and an inter-area routing tree. The trees are generated using the two separate databases.

The within-area routing tree for a Border Node is computed in a similar manner to that done by the internal node except it does not include areas. Once again the tree is generated from the information contained in the within-area database using the SPF algorithm.

The inter-area routing tree is computed using the Border Node's inter-area database (for border nodes outside the area) and its within-area routing tree (for border nodes within the area). The inter-area tree is the SPF-generated tree for the network consisting of the Border Nodes, their inter-area actual links and their intra-area delays.

3.3.3 Packet Processing Scenarios

3.3.3.1 Packet Origination from an Internal Node

After packet entry, the node checks the destination of the packet. If it is in the same area the packet is routed using the node's routing table, in accordance with the SPF algorithm. However, if the destination is in another area the node routes the packet using the destination area's entry in the routing table, again in accordance with the SPF algorithm.

3.3.3.2 Packet Origination from a Border Node

After packet entry, the node then checks the destination of the packet. If the destination is an internal node in the same area, the packet is routed in the same manner as for the internal node above. If the destination is a border node (in the same area or in a remote area), the packet is routed using the Border Node's inter-area routing table according to the destination Border Node entry. If the destination is an internal node in another area, the packet is routed to the Border Node in the destination area which minimizes the path length.

3.3.3.3 Packet Continuation at an Internal Node

Internal nodes may receive in-transit packets with destinations either within or outside of the area. When an internal node receives a within-area packet from one of its neighbors it checks to see if it is the recipient. If so, the packet is delivered. Otherwise, the node forwards the packet to the next node on the path to the destination node, according to its routing table.

When an internal node receives a inter-area packet it forwards it to the border node designated in the packet header as the "next border node", according to its routing table. (The need for two levels of addressing for inter-area packets is discussed below in Section 3.4.)

3.3.3.4 Packet Continuation at a Border Node

Border nodes may receive in-transit packets with destinations either within or outside of the area. A Border Node handles a within-area packet in the same manner as an internal node, forwarding it to the next node when it is not the recipient.

When a Border Node receives an inter-area packet it first checks to see if it is the recipient. If so, the packet is delivered. Otherwise, it checks the destination to see if it is outside the present area. If so, it forwards the packet according to its inter-area routing table, noting in the packet header the "next border node" along the path. If the packet is within the same area, the packet is "reduced" to a within-area packet and the within-area routing table is used. ("Reduction" of a packet simply replaces the two-level address with a single, within-area address.)

3.4 Addressing

No explicit requirements are placed upon the addressing scheme to distinguish and identify nodes in the network, except that each node requires a unique identification. (In other words, a logical addressing approach is specified.) Specific node processing (as discussed above) of both user data and routing update messages does depend upon the status of the originator--internal node versus border. However, rather than encode this distinction in the address, our design relies upon the look up table as the source of information about the status of the originator.

Two levels of addressing in packet headers are necessary in area routing to accomplish loop-free second level routing. In both within-area and inter-area addressing the originating node's name and the destination node are specified. In addition to the primary source and destination, a field is added to the inter-area packet header to specify the next Border

Node to route to on the path to the final destination. A Border Node sending or forwarding a packet to a node in another area must obtain the next Border Node to route to from its inter-area routing table and include this information in the "next Border Node" field of the packet header.

Since inter-area update messages are sent separately from intra-area updates, a node's Border Node-to-Border Node routing information may be inconsistent with other area's first level routing information at internal nodes. The following example illustrates the need for a second level of addressing:

> In Figure 3-1, if e_3 wants to route information to f_2, it would take the Border Node hops: e_3 / g_1 / g_2 / f_5 / f_1; since e_3 reports the least delay to area F entering at Border Node f_5. Now assume this path was taken but the next border node information was omitted from the header and delay across the g_3 - g_2 link increased to 8. When e_3 routes to g_3 via g_1, g_3 would realize that his shortest path to area F is via e_3 border node f_1 and route the packet back to e_3. This would result in a loop between e_3 and g_3 which would continue until the Border Node updates for area G are sent to other Border Nodes in the network which reflect the change in delay from g_1 to g_2. Therefore, it is important for Border Node g_1 to explicitly state Border Node g_2 as an intermediate destination on route to node f_2. Thus justifying the need for the second level of addressing.

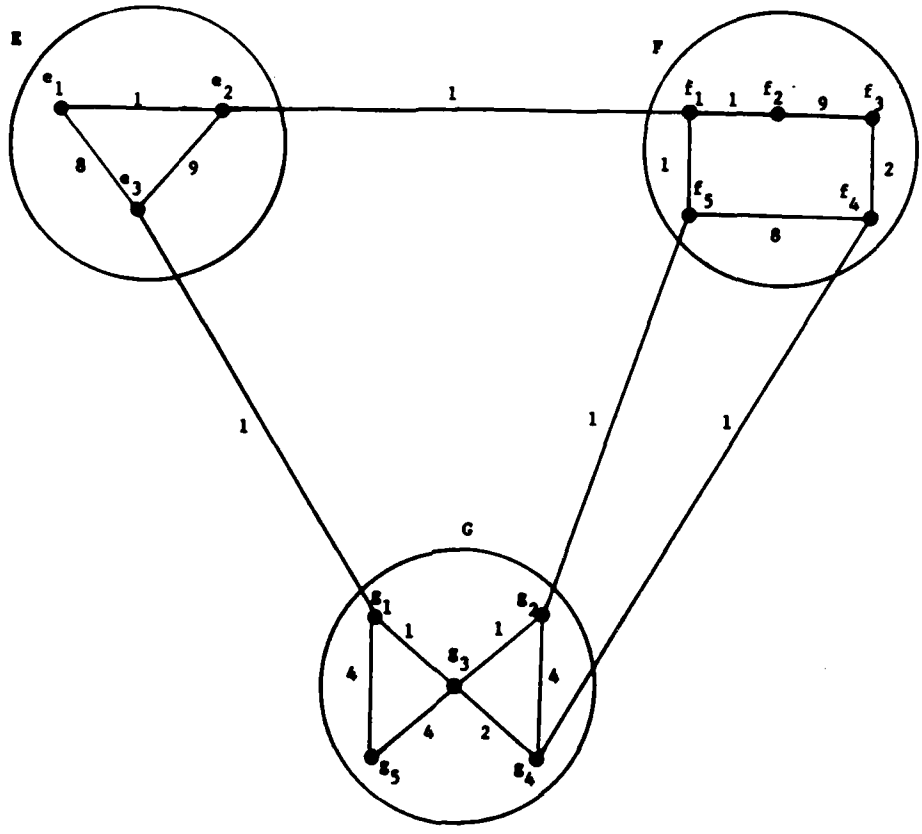


Figure 3-1. Area Routing Example

3.5 Update Messages

Internal nodes send local updates to all nodes within the same area consisting of its distance to all other nodes in the area. (See Figure 3-2.)

A border node sends within-area updates to nodes within its area consisting of its distance to all nodes within its own area plus distance to all remote areas. (See Figure 3-3.) Border nodes send inter-area update messages to other border nodes consisting of area information concerning delays across its area to each other local border node and delay to each neighboring border node in another area (i.e. delay across

interarea link). Border nodes obtain the information concerning what nodes to route inter-area updates to (i.e., which nodes are Border Nodes) from the Node Directory. (See Figure 3-4.)

Neighbor	Distance
e_2	..
e_3	..

Figure 3-2. Example of a Within-Area Update of Internal Node e_1 of Figure 3-1.

Neighbor	Distance
e_1	..
e_3	..
F	..
G	..

Figure 3-3. Example of a Within-Area Update from Border Node e_2 of Figure 3-1.

Neighbor	Distance
e_3	..
f_1	..
f_5	..
f_4	..
g_1	..
g_2	..
g_4	..

Figure 3-4. Example of Inter-area update from Border Node e_2 of Figure 3-1.

3.6 Area Formation Techniques

Areas are defined initially by data that are loaded into each of packet switching node's Node Directory. The node directory is consulted continually during the use of the logical addressing scheme prescribed here. This also gives the network the capability to have areas redefined during the course of normal operations, as a result of a privileged down-load of a new Node Directory to each node.

In order to avoid complications of asynchronous incorporation of new Node Directories, including possible misrouting of some copies of the Node Directory, the downloading process shall incorporate a mandatory wait period of storage before the new Node Directory is implemented. This permits distribution of the Node Directory under an existing if not stable set of routes.

As noted in Section 2.4, the principles of optimal area definition are only partially understood. There is understanding of the effects of area definition within the context of routing table size, but not within the context of expected network performance. Therefore, there is no current guideline for the need for area definition in response to anything but changes in network topology. Even here, it is clear what is to be expected when area redefinition is attempted in response to non-serious topology changes (ones that have not resulted in partitions).

As more is learned about network performance as a function of area definition, a network monitoring center can monitor node-pair traffic demands and periodically redesign the areas and retransmit the Node Directory to all nodes.

In some instances, loss of network nodes can result in partition of areas. This definitely requires redefinition of areas and a subsequent

rebroadcast of Node Directories. Therefore, this capability must be included in an authorized network control center.

3.7 Non-Use of Global Routing Nodes

The use of Global Routing Nodes was raised as a design issue in section 2, largely because of the historical use of GRNs in hierarchical organizations of Packet Radio Networks. No use of GRNs is proposed here, for two reasons. First, in the event of network topology changes, there is little that a GRN can accomplish in a terrestrial network. In a Packet Radio Network, a GRN can direct the establishment of new node-to-node links; similar opportunities do not exist in terrestrial networks. Second, there is little need to have a set of more powerful, centralized GRN nodes to manage less capable nodes in the DDN, where each node has ample computing capability.

3.8 Update Exchange Protocols

The principles behind the exchange of update messages in the current flat routing scheme have proved reliable enough and efficient for operation in the current DDN; this design extends their application to area routing.

For within-area routing, update messages among nodes within the same area are exchanged in accordance with the simple flood discipline: updates generated by a node are transmitted on each of its intra-area links; the recipient checks to see if the message represents new information within the current sequence number epoch; if not, the message has been received previously and so is discarded; if so, the recipient's database is updated in accordance with the new information.

The frequency of new update messages is determined in accordance with the current implementation, with 10 second averaging and a steadily decreasing threshold of determination of significance that becomes zero

after 5Ø seconds. Therefore, the minimum update period is 1Ø seconds, while the maximum is 5Ø seconds. These are tunable parameters and must be adjusted to the rate of network topology changes through link failures, recoveries and the like.

For inter-area updates a similar principle is applied, except that the discipline is applied only among border nodes. Inter-area routing updates are specifically identified and so are not flooded among the internal nodes, even though they are transmitted to other border nodes via internal nodes.

Border nodes monitor both the delays on direct links to other border nodes and the SPF-calculated delays to border nodes in the same area. As with the monitoring of delays by internal nodes, the averaging interval is 1Ø seconds. However, the threshold for significant change in the calculated delays is larger than the threshold for a single link, by a factor of the number of nodes in the *minimum* path to the border node.

When a border node detects a significant change, it sends *an* update message to each border node to which it is directly connected and to each border node in the same area. When a border node receives an inter-area update message, it checks to see if the message represents new information within the current sequence number epoch; if not, the message has been received previously and so is discarded; if so, the recipient's database is updated in accordance with the new information.

3.9 Response to Network Partitions

An area partition can be recognized by nodes within the area as failures to find any path to one or more nodes in the area via the SPF algorithm. This condition, when detected, is reported to an authorized network control center. The control center will receive several such

messages from nodes within an area and will make a response based upon their collective content. The control center will define new areas that encompass all nodes known to be operational, and it will broadcast a new area definition.

3.1Ø Examples of Area Routing

Three cases that illustrate area routing examples are described below. Reference is made to Figure 3-1.

1. Within-Area Routing: Internal Node to Border Node. Routing to a node within an area would be accomplished using within-area routing tables and would be amongst only nodes within the area. Routing from e_1 to e_3 would be e_1 / e_3 instead of going outside the area along a more optimal route. Frequently, staying within an area would be optimal.

2. Inter-Area Routing: Internal Node to Remote Border Node. Routing from internal node e_1 to g_1 would take the path: e_1 / e_2 / f_1 / f_5 / g_2 / g_3 / g_1. This is accomplished in the following manner. Node e_1 looks up g_1 in the Node Directory to find the destination area. It then routes the packet according to its routing table using the destination area entry. The shortest path to area G from e_1 is to "exit" through Border Node e_2 so the packet is routed to e_2. Once at Border Node e_2, g_1 is looked up in the Node Directory and found to be a Border Node. Since e_2 has an entry for every border node in the network, it routes the packet directly to g1 according to the path recorded in its inter-area routing table.

3. Inter-Area Routing: Internal Node to Remote Internal Node. Routing from internal node e_1 to internal node g_3 would be the same as above in approach except once at "exit" Border Node e_2, it would route to whichever Border Node in area G that minimizes the path. Once at the entrance into area G via Border Node g_2, within-area routing would be used to route to g_3. The resulting path would be: e_1 / e_2 / f_1 / f_5 / g_2 / g_3.

4. Inter-Area Routing: Internal Node to Remote Border Node (Special Case). In routing from an internal node to a border node in a remote area a special case of nonoptimality could occur as follows: Consider the process that would be followed if a packet was to be routed from internal node g_5 to border node f_4. g_5 would look up the destination in its Node Directory and find that it is in a remote area. It would then route the packet to the border node in its own area which reports the least delay to the remote area. In this case it would be g_2, using the path g_5 / g_3/ g_2. Once the packet reaches g_2, however, g_2 looks up destination f_4 in the Node Directory and finds it is a border node. Since g_2 has an entry for every border node in its inter-area routing table, it is able to

route the packet directly to f_4 using the path recorded in the table. In this case the the path would be g_2 / g_3 / g_4 / f_4. The packet must pass through a previously visited node, g_3, along its most optimal path to f_4. Any other path from g_2 would result in a longer delay. This case is an example of the nonoptimality that can occur since the internal nodes contain less information then the border nodes concerning the configuration of the remote areas.

5. Within-Area Routing: (Special Case) When routing a packet from a source node to a destination node that are both in the same area a special case of nonoptimality could occur. Consider the routing procedure when a packet is to be routed from f_2 to f_4. When routing from an internal node to a border node (or vice-versa) in the same area, the path the packet takes is restrained to the area since limited information is retained by the nodes concerning the configuration of remote area. Thus, the packet would be routed on the path f_2 / f_1 / f_5 / f_4 for a total delay of 10. However, the most optimal path, which would be used in a flat routing scheme, would be found going out of the area along the path f_2 / f_1 / f_5 / g_2 / g_4 / f_4 for a total delay of 8. This case is an example of the nonoptimality that could occur in an area routing scheme vs. a flat routing scheme.

3.11 Estimation of Control Traffic

The costs and benefits of this (and other) area routing scheme are determined by the savings in control traffic versus the generation of non-optimal routes. This section present calculations of update traffic under the area routing design. For update messages by internal nodes, the size of an update depends on the number of links attached to the node and equals:

$$136 + C*16 \text{ bits}$$

where C is the node degree. 136 bits is the amount of required packet header bits.

The following assumptions about the update frequencies may be used:

1. Within-area updates are generated, on the average, once every 30 seconds.

2. Border Node updates are generated, on the average, once every 30 seconds.

as discussed in Section 3.8.

The total traffic due to internal updates is the number of internal nodes in a single (or average) area times the average message size per internal node. The average message size is 136 + the average branching degree * 16 bits. The flooding algorithm causes, to a close approximation, each node's update message to flow once on every area link.

Border nodes generate additional update messages within an area about their estimate of distances to remote areas. In analogy with link updates,, 16 bits per area are required. Within a single area, the contribution from border nodes is slightly larger than that from internal nodes.

Border node updates flow only on selected links in the network: border-node-to-border node direct links and the virtual paths through areas. Each border node sends an update message of length 136 + (length of second level address) + 16 * number of "neighbor" border nodes. The total border node inter-area traffic is the number of border nodes times the average message length per border node. However, this second type of update traffic does not flow on all network links, but only on inter-border-node routes.

The following connectivity assumptions are made for a typical large network, consisting of 1000 nodes organized into 10 areas of 100 nodes each:

1. internal nodes have an average of three neighbors;
2. border nodes have an average of three internal neighbors and one neighbor in another area.

Therefore, internal updates are an average of 184 bits, while border node internal updates are 184 + 9 (other areas) * 16 = 328 bits. Border node inter-area updates are 136 + 9 (other internal border nodes) * 16 + 1 (inter-area link) * 16 = 296 bits. Consequently, traffic flowing within a

typical area due to within-area updates is 90 (internal nodes) * 184 + 10 (border nodes) * 328 , per 30 seconds = 661 bits/second. The traffic flowing over border node to border node routes = 100 (border nodes) * 296 bits, per thirty seconds = 986 bits per second. However, this traffic is distributed only over a fraction of the network links. Assuming this fraction is one in four, the update traffic may be calculated as 661 + 246.5 = 907.5 bits/second. The comparable bits per second for flat routing, presented on page 5, is 613 bits/second. In terms of percentage of bandwidth of 56 KBPS trunks connecting the nodes, these represent 1.6% and 11% respectively.

For a smaller network, the 19 node ARPANET model used in Section 5, the calculations differ. Again the branching degree per node is an average of three, but there are two areas: one with 10 and one with 9 nodes. In the first area, there are 5 border nodes, and there are 4 border nodes in the second. (This is a different border node to internal node ratio.) Consequently, the Within-Area Update Message Size = 184 bits for an interior node, and 200 bits for a border node's update covering its single inter-area link. For a border node's inter-area update, the message length is 136 + 4 (other border in area) + 1 (border node directly linked) * 16 = 216 bits.

Within the 10 node area, the internal update traffic (per link) is 5 (internal nodes) * 184 + 5 (border nodes) * 200, per 30 seconds = 1920/30 = 64 bits per second. The update traffic over the border node direct links is 1 (border node per unidirectional link) * 216, per thirty seconds, or 6.6 bits per second. A total of 5 border nodes contribute 216 bit messages every thirty seconds over the shortest paths between border nodes. This is not the equivalent of flooding. For simplicity, we assume that only one in

four links are involved. Therefore, we estimate that the per link traffic to support border node inter area updates is (1/4) * 5 * 216, per 30 seconds = 9 bits per second. The comparable figures for flat routing overhead traffic volume under flat routing is 116 bits/second. By limiting the area size, a reduction in routing update traffic is realized.

The primary effect in reducing the control traffic requirements in the area routing scheme proposed here (and in similar ones) is in having fewer nodes that use the flooding procedure. The contribution of this effect is to reduce control traffic by a factor of 1/(number of areas). A secondary effect will add traffic: border nodes must exchange their topological updates. The design presented here does not flood these messages, thereby diminishing their contribution to the control traffic. (Actual experience could later dictate against this choice, if a non-flooding discipline proves to be of insufficient reliability for the exchange of border node topological information.) The contribution from border nodes will be dependent upon the particular topology that results from the area definitions. The resulting reduction in control traffic is 1/(number of areas) + (small additive term for border node contributions).

3.12 Design Drivers and Alternatives

This section discusses factors related to the characteristics of this design for area routing. First, factors that would influence designs in general toward the one presented here are discussed. Second, some design alternatives are presented.

3.12.1 Design Drivers

This design for area routing has been strongly influenced by several driving factors related to the current designs and directions for the Defense Data Network. First, the current size of the DDN dictates that any

radically different approach to routing would be extremely expensive. This expense would be likely to overshadow potential benefits that could be demonstrated for a radical scheme. Instead, area routing represents a more evolutionary approach to a routing algorithm. Second, the link topology for the DDN is mandated by requirements for robustness (survivability). Given the branching degree of each node and the large number of nodes, connectivity in the network can be maintained despite the failure of a large number of nodes and links. A new routing algorithm must operate within this same link topology and be able to maintain network connectivity competitively with the current scheme. This requirement argues strongly against purely hierarchical routing schemes. Third, a routing algorithm design must promise nearly optimal paths, in order that the gains from reduced control traffic not be lost to reduced overall network bandwidth that would result from too many non-optimal paths being used.

These factors have strongly influenced the design for an area routing algorithm. Specifically, the organization into areas permits the existing node and link topology to be retained and fully utilized. Also, the differentiation of node function among the levels (intra-area and inter-area) has been kept to a minimum. Finally, the design uses a more detailed representations of areas, compared to its counterparts in purely hierarchical schemes. This permits more nearly optimal paths to be obtained, compared to cases where very little information would be used to describe areas.

Routing in the DoD Internetwork is an example of purely hierarchical routing, in which very little information is used to describe the "lower" level (i.e., the particular subnetworks) to the "upper" level (i.e. routing performed among gateways. To a gateway, a particular subnetwork appears

simply as a monolithic "hop" along a datagram route. Although the Internetwork R and D community agrees on the need for a suitable distance metric, none is currently in used, so the distance across any network is simply "one hop". In addition, there is no differentiation regarding the distance across a particular subnetwork with respect to the direction of crossing. (In broadcast networks, this is not a factor.)

These identifiable flaws in the Internetwork routing scheme motivate their treatment in the area routing scheme. Since area routing operates in a single network among mutually cooperative and trustful nodes, a distance/delay metric can be used by all. This distance metric is used to express the delays between entry and exit points for the areas. Therefore, the higher-level routing can distinguish better when to use which areas for an inter-area route.

3.12.2 Design Alternatives

This section identifies and discusses specific design alternatives for the area routing scheme presented above. We found no strong reasons for incorporating any of these design alternatives into our area routing scheme; in each case, the specific utility is counter-balanced by several disadvantages.

3.12.2.1 Border Node Sharing Multiple Areas

In this approach the organization of the network would be similar to the one used in the above design except that a border node could be contained within more than one area. This would eliminate actual interarea links between the "shared" areas but would retain connectivity between areas that do not share a border node.

This approach would give a border node more insight into at least one more area (possibly more depending on the number of areas it shares),

thus yielding more optimal paths between the shared areas. However, this approach has several drawbacks. First, the border node must contain more information since it would need routing tables and databases for each area (within-area) or larger routing tables and databases (inter-area). Second, border nodes would have to be able to discern what update information to send to internal nodes (i.e., sending only the information concerning the internal node's area). Finally, classifying types of nodes and updating the databases would be more complicated.

3.12.2.2 Internal to Border Node Relation.

In this approach, each internal node would be related to a particular border node, inherent in the nodes' addresses. The internal node would route all packets leaving the area via its related border node. This approach is currently being investigated by SRI in the survivable Internet program.

Using this approach, it would not be necessary to keep information concerning routing to remote areas within the internal nodes. All traffic leaving the area would be routed to the related border node, and the border node would contain the necessary information to continue the routing process. However, if a border node goes down, all of its internal nodes must be reassigned to another border node (i.e. their addresses changed to reflect the new border node). If a link goes down which connects a border node to another area, then routing to only that border node would cause information to be lost or at best it would have to be rerouted to another operable border node. Finally, if an internal node is not given a choice of "exit" border nodes dependent on the destination area, then a suboptimal path may result.

4. Simulation and Analytical Results

This section presents the results of an experimental study of the efficiency of the area routing algorithm described in Section 3. The experiment is a static one that assesses the non-optimality of routes developed under area routing, assuming completely synchronized nodal routing tables, particular area organizations, and several different sets of semi-random delays on links.

4.1 Method

Initial network configuration were chosen based upon a 19-node past ARPANET configuration and a 31 node randomly-generated network configuration. Figure 4-1 illustrates the 19 node configuration.

Hypothetical, randomly distributed delay measures for each directional link were generated by 1) generating a random number between Ø and Ø.9 to represent the link utilization, and 2) calculating the resulting expected "time in the system", assuming Markov arrival and service processes. "Time in the system" refers to the expected time per packet required in a transmission queue plus the time required for actual transmission. The basic time unit is chosen as the average time to transmit a packet. Therefore, the delay units may be regarded as "packet transmission times." The purpose in generating delay measures is not so much to represent expected link delays as to present variations to the routing algorithm, forcing it to pick other than shortest hop count routes in these representative configurations. Since a set of generated link delays represent a potential instantaneous delay set, they are termed "snapshots" below.

From each network configuration and each routing method, a "universal" routing table was calculated. That is, complete

synchronization of nodal routing information was implicitly assumed. Based upon the single "universal" routing tables, best routes were calculated for every source/destination pair in the network, under both flat routing (i.e., shortest path based upon Dijkstra's method) and upon area routing (as described in Section 4 above).

Under this approach, both the path lengths and the simulated total path delays on the network routes can be evaluated easily for several hypothetical "snapshots" of network link delays, avoiding biases that could arise from examining only a sample case. For each network configuration, 3 hypothetical snapshots were generated, representing sets of link delays (in units of packet transmission times) as might be seen by the respective routing algorithms.

4.2 Results

4.2.1 Delays Under Flat Routing

Three different snapshots for the each of the two network configurations were used in calculation of optimal routes between each distinct pair of end points in the network. Tables 4-1 A and B below reports the average hop count and its variance, and the average path delay (in "packet transmission" times).

TABLE 4-1A. PATH LENGTHS AND DELAYS FOR FLAT ROUTING IN A 19 NODE NETWORK

Delay Set #	Avg. Hops	Variance	Avg. Delay	Variance
1	3.81	1.61	4.62	4.61
2	4.Ø9	2.61	4.95	6.Ø4
3	3.8Ø	1.66	5.42	7.52

TABLE 4-1B. PATH LENGTHS AND DELAYS FOR FLAT ROUTING
IN A 31 NODE NETWORK

Delay Set #	Avg. Hops	Variance	Avg. Delay	Variance
1	4.97	3.3Ø	7.72	16.4
2	5.59	5.8Ø	7.74	17.1
3	5.62	6.77	7.91	18.7

Note that the variances result from the inherently differing optimal path lengths over all end pairs in the network; some are close and some are far. There is greater variance in the delays than in the path lengths due to the additional randomness in the link delay generation.

4.2.2 Delays Under Area Routing

Three different snapshots for the each of the two network configurations were used in calculation of area routes between every pair of end points in the network. Tables 4-2, -3, -4 and -5 below reports the average hop count and its variance, and the average path delay (in arbitrary units), for three different area configurations. Hagouel's "Version 3" algorithm (See [HAGØ83], Chapter 4) was used to cluster the network into areas containing either a maximum of 1Ø, 5 or 3 nodes for the 19 node network, or maximum sizes of 4, 7, 12 and 16 for the 31 node network.

TABLE 4-2A. PATH LENGTHS AND DELAYS FOR AREA ROUTING IN A 19 NODE NETWORK

MAXIMUM AREA SIZE: 3 NODES

Delay Set #	Avg. Hops	Variance	Avg. Delay	Variance
1	3.82	1.60	4.63	4.62
2	4.08	2.62	5.01	6.14
3	3.77	1.62	5.56	8.28

TABLE 4-2B. PATH LENGTHS AND DELAYS FOR AREA ROUTING IN A 31 NODE NETWORK

MAXIMUM AREA SIZE: 4 NODES

Delay Set #	Avg. Hops	Variance	Avg. Delay	Variance
1	5.00	3.41	7.98	17.1
2	5.56	5.68	8.01	19.3
3	5.61	6.77	7.80	19.1

Examination of these data show very similar path lengths, delays and variances to those obtained from simulating flat routing. This is born out by comparing the list of routes generated; they are identical under both schemes in nearly every case, because under this particular area organization, only one node (out of the nineteen) is a true interior node. The routes are generated optimally among the preponderance of border nodes. The minimal difference observed in the delay averages are the consequence of only 1 or 2 different routes calculated under the area scheme with three-node areas.

Tables 4-3 and 4-4 below show area routing results calculated under areas with larger maximum sizes. The difference in average delay, compared to flat routing is more noticeable, even though the respective variances are still large. Direct comparison of routes generated shows higher

frequencies of non-optimal routes with the larger areas, due to restrictions against choosing an outside route to a destination within the same area.

TABLE 4-3A. PATH LENGTHS AND DELAYS FOR AREA ROUTING IN A 19 NODE NETWORK

MAXIMUM AREA SIZE: 5 NODES

Delay Set #	Avg. Hops	Variance	Avg. Delay	Variance
1	3.85	1.75	4.72	5.11
2	4.Ø7	2.53	5.14	7.33
3	3.84	1.94	5.99	12.5

TABLE 4-3B. PATH LENGTHS AND DELAYS FOR AREA ROUTING IN A 31 NODE NETWORK

MAXIMUM AREA SIZE:7 NODES

Delay Set #	Avg. Hops	Variance	Avg. Delay	Variance
1	5.Ø2	3.57	8.25	22.Ø
2	5.52	5.52	8.Ø6	19.7
3	5.81	8.56	8.3Ø	21.6

TABLE 4-4A. PATH LENGTHS AND DELAYS FOR AREA ROUTING IN A 19 NODE NETWORK

MAXIMUM AREA SIZE: 1Ø NODES

Delay Set #	Avg. Hops	Variance	Avg. Delay	Variance
1	4.Ø4	2.37	4.91	6.Ø3
2	3.97	2.12	5.53	1Ø.9
3	3.84	2.23	5.87	9.71

TABLE 4-4B. PATH LENGTHS AND DELAYS FOR AREA ROUTING IN A 31 NODE NETWORK

MAXIMUM AREA SIZE: 12 NODES

Delay Set #	Avg. Hops	Variance	Avg. Delay	Variance
1	5.Ø3	3.46	7.84	17.2
2	5.65	5.99	7.83	17.3
3	5.68	7.Ø1	7.95	18.9

TABLE 4-4C. PATH LENGTHS AND DELAYS FOR AREA ROUTING IN A 31 NODE NETWORK

MAXIMUM AREA SIZE: 16 NODES

Delay Set #	Avg. Hops	Variance	Avg. Delay	Variance
1	5.Ø4	3.44	7.84	16.9
2	5.64	5.95	7.78	17.2
3	5.75	7.21	8.Ø1	19.3

4.2.3 Direct Comparison of Flat and Area Routing

Table 4-5 presents a side-by-side comparison of the average delays calculated for the 19 node network, under three different randomly generated link delay sets, and under flat, 3-node area, 5-node area and 1Ø-node area routing. This table makes more evident the small but consistent differences in the delays of area routes over flat routes.

TABLE 4-5A. COMPARISON OF FLAT AND AREA ROUTING
IN A 19 NODE NETWORK

AVERAGE DELAYS (ARBITRARY UNITS)

Flat Routes	3-Node Areas	5-Node Areas	10-Node Areas
4.62 (0%)	4.65 (.6%)	4.85 (4.9%)	5.08 (10%)
4.95 (0%)	5.05 (2 %)	5.41 (9.3%)	6.03 (21%)
5.42 (0%)	5.57 (2.8%)	6.03 (11.2%)	6.27 (15%)

TABLE 4-5B. COMPARISON OF FLAT AND AREA ROUTING
IN A 31 NODE NETWORK

AVERAGE DELAYS (ARBITRARY UNITS)

Flat	4 Nodes	7 Nodes	12 Nodes	16 Nodes
7.72 (0%)	7.83 (1.4%)	8.28 (7.2%)	7.85 (1.6%)	7.85 (1.6%)
7.74 (0%)	8.02 (3.6%)	8.09 (4.5%)	7.84 (6.3%)	7.78 (0.5%)
7.91 (0%)	8.17 (3.2%)	8.60 (8.7%)	7.95 (0.5%)	8.01 (1.2%)

(Percentages above optimal routing are shown in parentheses.)

4.3 Discussion

4.3.1 Flat Versus Area Routing

The results show that the average path length and average delay are very weak discriminant between flat routing and area routing for the network models examined here. Both path length and delay are slightly larger for area routing than for flat routing, but the difference is small compared to the variance attributable to the inherently different paths between network node pairs under either scheme. The consequence of this finding is that there is not a severe penalty to be paid, using area routing, in terms of non-optimal routes. This finding is consistent with

that of Callon and Lauer [CAL85], who failed to note any strikingly different path lengths under hierarchical, flat and semi-hierarchical routing.

In each evaluation there are either 19*18 = 342 or 31*30 = 930 distinct paths formed, so that case-by-case examination is very tedious. Nevertheless, scanning paths created by the two methods showed that frequently an optimal path could be taken under area routing. In the model examined here, non-optimal paths arose most often from restriction to within area routes, when a better route could be obtained by using a neighboring area. The percentages of path delay above optimality shown in Table 4-5A reflect the frequency of such routes in 3- versus 5- versus 10 maximum size areas. That is, in 3-node areas, practically every node can be a border node, enabling optimal routes to be found very often. In 10 node areas, more paths must be confined to an area.

Table 4-5B shows larger areas (16 node maximum) with more nearly optimal path delays, compared to 4- and 7-node maximum size areas. This is expected, since both the smallest size areas (1 node) and the largest size areas (e.g., 31 nodes) could be expected to have optimal performance under area routing. In both extreme cases, nodes have complete topological information upon which to base routing decisions.

4.3.2 Implications for Area Design

These experiments enumerated paths between network nodes that would result by the two different algorithms using delay-based metrics. The actual network performance would need to be assessed according to the traffic load on each of those paths. Depending upon whether more traffic was sent on the better or worse paths, during an arbitrary time period, the better or worse network performance would be. This does suggest definition

of areas to encompass major traffic flows, so that they can more often be accommodated over optimal paths. However, there is also an argument against this approach--encompassing heavier traffic within an area could lead to premature congestion because of restricted load balancing opportunities. These observations underscore the point raised in Section 2.4 above, that the network performance implications of area definitions are not completely understood.

4.3.3 General Conclusions

The experiments demonstrate that in these cases at least, the penalty in terms of non-optimal routes is small for area routing (e.g., in the range of only 2 - 10% worse than optimal). However, this does not consider weighting by actual path usage, which could change the penalty degree.

However, for the case of the 19 node network, the relative area routing penalties, on the order of 5% as shown in Table 4.5A, outweighed the relative update traffic savings (116 bits per second with flat routing versus approximately 80 bits per second with area routing)--1.2% versus 0.8% of 56000 bits per second. Clearly, these are guarded observations, because of the limited nature of the network model. However, they do not clearly indicate any significant technical payoffs, relative to technical costs, through area routing.

These interpretations are generally applicable to area routing, rather than only to the particular algorithm presented in this report. The gains in control traffic are generally a function of the area size and border node population, as are the degrees of path non-optimality. Minor differences may be obtained among area routing models, but the "ball park" estimates are likely to be consistent. The figure for projected DDN

configurations is use of 11% of the bandwidth of 56000 network links to support control traffic for flat routing. On the other hand, delays due to non-optimal paths may be on the order of 5%, as shown in Tables 4-5A and B. (Also, delays can be expected to be more non-optimal in area structures that do more to reduce control traffic.) Under these conditions, the use of area routing would result in only marginal bandwidth gain over flat routing.

5. Conclusions

In this report, a design for an area algorithm has been presented in Section 3 and then used in support of a critical analysis, in Section 4, of the potential utility of area routing for use in large networks. Specifically, the design was used to calculate paths in model networks, and these paths were compared to optimally generated ones, to assess the potential for non-optimal routing. The findings motivate some doubt with regard to the utility of area routing. Although only a limited potential for non-optimal routing was observed, this potential was still comparable to the reduction in control traffic that could be realized through area routing. This finding does presume the use of 56000 and higher bandwidth links. If links of significantly lower bandwidths were assumed, the benefits of area routing would be more substantial. If links of significantly higher bandwidths were assumed, the benefits would be very questionable.

Overall, this report has addressed primarily the problems associated with conserving network link bandwidth in large networks. The general motivation for area routing and other proposals is the reduction of the amount of control traffic. This appears to be a feasible, low risk approach. Is it possible to achieve decreases in network delays without decreasing the control traffic? The Appendix of this report demonstrates that it is possible, using a multi-path routing strategy, to realize improved network performance without resorting specifically to control traffic reduction.

This study has also clarified several issues in the design of routing algorithms for large networks. The advantages of semi-hierarchical routing techniques, such as area routing, are improved representation

between the levels, which makes the decisions at each more nearly optimal. The higher level has a more detailed representation of the lower level--the areas, and it can choose better routes among several areas as a result. At the same time, there is economy achieved by sparing lower level nodes of having to maintain information about all other nodes. This principle gives area routing significant advantages over purely hierarchical schemes, such as DoD Internetworking. In the Internetworking approach, there is a strong separation between the levels--IP datagram routing and subnetwork routing. Neither is cognizant of the activities of the other. The non-optimal routes that can be developed under Internetworking are well known.

6. References

BAK83 D.J. Baker, "Distributed Control of Broadcast Radio Networks with Changing Topologies," Proceedings of IEEE INFOCOM83, pp. 49-55.

BBN86 A. Khanna, J. Seeger, "Large Network Routing Study Design Document," BBN report 6119, January 1986.

CAL85 R. Callon, G. Lauer, "Hierarchical Routing for Packet Radio Networks," BBN report 5945, June 1985.

DJIK58 Dijkstra, E. W., "A Note on Two Problems in Connection with Graphs," Numer. Math. Vol. 1, pp. 269-271: 1959.

GARE79 Garey, M. R. and Johnson, D. S., Computers and Intractability. A Guide to the Theory of NP-Completeness (W. H. Freeman and Co., San Francisco, 1979).

GAVI83 Gavish, R. and Hantler, S. L., "An Algorithm for Optimal Route Selection in SNA Networks," IEEE Trans. on Commun. COM-31: 1154-61 (Oct. 83).

GRAY79 Gray, J. P. and McNeill, T. B., "SNA Multiple System Networking," IBM Systems J. 18: 263-297 (1979).

HAGO83 Hagouel, J., "Issues in Routing for Large Dynamic Networks," Ph. D. Thesis, Columbia University, 1983.

HIL81 W. Hilal, M.T. Liu, "Guided-Adaptive Routing Techniques for Packet-Switching Networks," Proceedings of the Computer Networking Symposium, Dec 1981, pp. 65-74.

JAF84 J.M. Jaffe, A. Seagall, "Automatic Update of Replicated Topology Data Bases," Computer Communications Review, vol 14, num 2, pp. 142-148.

JAF85 J. M. Jaffe, SIAM J. Comput. 14(4): 875-999 (November 1985).

JAFF82 Jaffe, J. M. and Moss, F. H., IEEE Trans. on Commun. COM-30: 1758-1762 (July 1982).

KAH78 R.E. Kahn, S.A. Gronemeyer, J.Burchfield, R.C. Kunzelman, "Advances in Packet Radio Technology," Proceedings of the IEEE, vol. 66, num. 11, Nov. 78, pp. 1468-1496.

KLEI77 L. Kleinrock, F. Kamoun, "Hierarchical Routing for Large Networks," Computer Networks, 1977, volume I, pp. 155-174.

McQ80 John M. McQuillan, Ira Richer, Eric M. Rose, "The New Routing Algorithm for the ARPANET," IEEE Transactions on Communications, vol COM-28, num 5, May 1980, pp. 711-719.

SHA85 Nacham Shacham, J.J. Garcia Luna Aceves, "Analysis of Routing Strategies for Packet Radio Networks," Proceedings of IEEE INFOCOM85, pp. 191-302.

TYME81 Tymes, L. R. W., "Routing and Flow Control in TYMNET," IEEE Transactions on Commun. COM-29: 392-398 (1981).

YUM81a Yum, T-S. P., "The Design and Analysis of a Semi Dynamic Deterministic Routing Rule," IEEE Trans. on Commun. COM-29: 498-504 (April 1981).

YUM81b Yum, T-S. P. and Schwartz, M., "The Join-Biased Queue Rule and Its Application to Routing in Computer Networks," IEEE Trans. on Commun. COM-29: 505-511 (April, 1981).

Appendix A.
Guided Adaptive Routing for the DDN

The motivation for research of area routing schemes is to develop techniques for managing very large networks. It is thought that current approaches to routing will operate too inefficiently to provide sufficiently high throughput and low average packet delay to meet DDN requirements. It is not clear that area routing techniques offer the most promising avenue for coping with the management of large networks. As an alternative to area routing approaches, a "guided adaptive" technique, which uses precomputed alternate paths and a flat address space, has been developed.

Though differing significantly in detail from their algorithm, the multipath approach described below was inspired largely by Hilal and Liu's paper, "Guided-Adaptive Routing Techniques for Packet-Switching Networks." The important characteristic of separating topological from traffic updates is retained. Another similarity that is retained with Hilal and Liu's approach is that the algorithm does not freely recompute paths to adapt to high traffic delays, but instead utilizes precomputed alternate paths. Hilal and Liu refer to routines which adapt in this fashion as "guided adaptive" algorithms.

Hilal and Liu do not give a method for the calculation of alternate paths. The method used in this algorithm has similarities to a multipath algorithm suggested by BBN. In the BBN algorithm, alternate paths are constructed by reforming an SPF tree, excluding the most highly congested link. In this algorithm, alternative routes are generated by performing the SPF algorithm, excluding an adjacent neighbor.

Appendix A: Guided Adaptive Routing for the DDN

As with the ARPANET routing algorithm, the primary design goal of the algorithm described below is to minimize average packet delay, by avoiding links experiencing long queuing delays. Using the proposed approach, there is reason to believe that more bandwidth can be made available for data packets by separation of traffic information from topological messages. Since traffic levels are reported only as increases or decreases in discrete states, there is a possibility that the average length of an update packet can be reduced. Finally, there is a possibility that the following algorithm could result in lower processor demands, since route calculations are performed only in response to topological changes. However, processing cycles are not thought to be a limiting network resource.

A.1 Multipath Objectives

Nodes have only limited actions they take can in response to reports of high link utilization. Packets from attached hosts can be temporarily blocked from access to the network, while tandem packets can be rerouted, held in queues, or dropped.

Dropping packets is obviously very wasteful of network resources. Merely holding packets, though not as wasteful as dropping them, still results in long queuing delays. Furthermore, if the link utilization is the result of an imbalance between arrival and service rates of a resource, then all available memory for queuing packets will eventually be exceeded. Hence, the most viable alternative for coping with high link utilization is to construct an alternate path, if possible.

The most important characteristic of the secondary paths which are generated by this guided adaptive routing algorithm is that each has a different next hop than its corresponding primary path. Hence, if the

threshold is reached for a node to use a secondary path, then that node immediately begins making changes to network traffic flows. In other multipath schemes, there may be some nodes for which the decision to use secondary paths fails to make any difference in the next hops they choose. In these algorithms, for some nodes, the additional overhead of a multipath scheme is wasted.

A guiding rule which shaped the development of this multipath scheme is that the decision by a node to use a secondary path ought to make an observable difference in its behavior. Another assumption which guided the development of this multipath scheme is that in general it is a poor practice to use secondary paths to destinations only one hop away. If packets destined for an adjacent node are sent to some other node, then the alternate path is at least twice the length of the primary path. It would be much more efficient to divert traffic further upstream.

The multipath algorithm described below makes the simplifying assumption that all links in the network are of equal capacity. With some work, the algorithm could be extended to cope with network configurations which include links of varying capacity.

A.2 Algorithm Description

In the proposed algorithm, a node will be required to maintain information about a primary path to each of its destinations. In addition, for many, but not all, destinations, information will be maintained on a secondary path. For any path, it is assumed that the next hop, the hop count (i.e., path length) and a delay measure can be accessed by the routing procedure; dynamic adjustments to the delay measure are central to the algorithm, and will be described in detail below. The algorithm also requires knowledge of all links used in its primary and secondary paths.

Appendix A: Guided Adaptive Routing for the DDN

Given a link identifier, a node must have the ability to determine each of its primary and secondary paths which include that link.

In the proposed scheme, route computation is performed only at network initialization, and upon notification of changes to the network topology. The mechanism for such notification is not treated in any detail below, since it could be readily effected by flooding, as in the current ARPANET.

The computation of primary routes is to be performed by use of Dijkstra's SPF algorithm, much as in the current ARPANET. In the process of generating the SPF tree, for each destination the path's hop count is computed, and the links along that path are recorded. After the primary SPF tree is constructed, the secondary routes to destinations are derived. The technique for generating secondary routes is as follows:

For each adjacent neighbor:

generate a set of that neighbor's descendants from the SPF tree;

attempt to reattach those descendants to that portion of the primary SPF tree which does not include the subtree of the adjacent neighbor under consideration; in this process, no links incident to the adjacent neighbor may be used.

for those nodes which could be reattached, if the hop count for their candidate alternate path is no more than two greater than the hop count for their primary path, record the hop count, next hop, and included links for their secondary paths.

for the remaining, unattached descendants of the adjacent node under consideration, as well as for the adjacent node itself, indicate that no secondary path exists.

These routes are generated only at node startup and upon changes in network topology. Unlike the ARPANET, delays due to high congestion will not result in route recalculation. Instead, high delays will trigger use of secondary paths.

As can be seen, for some destinations, there will be no secondary paths. For example, there are no secondary paths to adjacent nodes. Another point to note is that no secondary path is more than two hops longer than the primary path.

The delay measures between two nodes are initialized to the length of a min-hop path between those nodes. These values will be increased if high utilization is reported on links included in their associated paths, and decreased if news arrives that the links have become less congested. A node uses the delay measures, together with link queue sizes, on a per packet basis to choose between the primary and secondary paths; they are not used, and need not be maintained, if there is no secondary path to a destination.

With exceptions to prevent packet looping, the decision between a primary and secondary path is made as follows:

```
If (primary route delay + primary route queue) <=
   (secondary route delay + secondary route queue)
then
      choose the primary path
else
      choose the secondary path.
```

Note that by using the above procedure, secondary paths will be used as soon as there is local evidence to indicate that packets on those paths would arrive at their destination with less delay than packets taking the primary path. This splitting of traffic would occur most beneficially if the primary and secondary route delays and next-hop queue sizes to a particular destination were equal. In this event, if some number of packets to that destination were resident at a particular node, then they would be alternated between the primary and secondary routes.

Appendix A: Guided Adaptive Routing for the DDN

As alluded to above, a node adjusts the route delay values according to reports of changes in link utilization. These update messages are triggered by queue sizes crossing predefined thresholds; in the algorithm as implemented in our simulation, queue sizes are divided into three regions, corresponding to low, medium, and high utilization levels. Update messages reporting changes in queue sizes are sent out on all links other than the link whose queue size changes state. The update messages are then propagated by flooding.

To avoid oscillation and too high a volume of update traffic, the thresholds for transitions from a lower to a higher utilization must not be the same as those for transition from the higher to the lower state. For example, if a node were to transmit an update indicating a transition from low to medium link utilization when the link's outbound queue grew to a length of seven, then it should not report a transition to low utilization until the queue dropped at least to five and perhaps even to four or less.

Separating the "transition up" from the "transition down" thresholds will induce stability, via hysteresis, into the system. Determining a suitable width to allow between transition up and transition down thresholds, as well as determining optimal queue length sizes to use for thresholds, is a matter for further research.

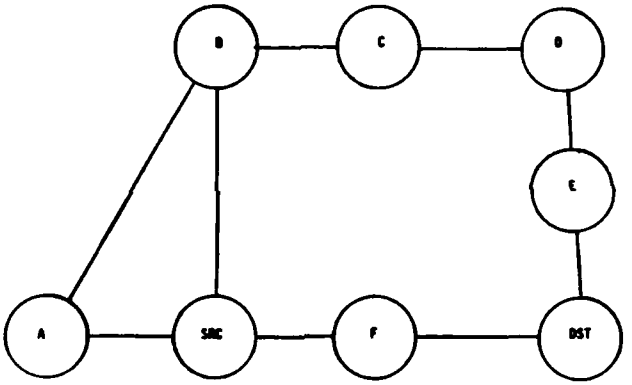
Nodes are to adjust their estimations of delay to various destinations based upon the receipt of link utilization updates. The adjustment is related to the level of utilization. For example, assume that the transition from a low to a medium utilization level is set to occur when a queue grows to a length of seven. In that case, upon receipt of an update which reports that a link has made such a transition, then the delay estimates for all paths which contain that link would be increased by

seven. If at a later time, an update is received which reports the link transitioning back to a low utilization level, then the delay estimates for each path containing that link would be reduced by seven.

A.3 Avoiding Routing Loops

If the decision procedure for choosing between primary and secondary paths is as simple as that described above, then long term traffic loops can result. For example, in the configuration shown below, assume that a packet at SRC had node DST as a destination. A routing loop would result if:

SRC chooses its secondary path to DST, transmitting to B.
B chooses its secondary path to DST, transmitting to A.
A chooses its primary path to DST, transmitting to SRC.



Network Configuration with Possible Routing Loops
Figure A3-I

The stability of the routing loop would depend upon congestion levels throughout the primary and secondary routes from the nodes in the loop to DST; there is no reason to assume that the loop would be transient.

One approach to avoiding such loops would be to constrain secondary paths so that no arbitrary sequence of primary and secondary routes to a single destination could result in a routing loop. This approach has two

drawbacks, however. It is computationally prohibitive, as it requires the generation of all primary and all candidate secondary paths between each possible source and destination. Also, the set of secondary paths is severely reduced. Using our previous example, in many cases, for node B to transmit a packet to DST, path (B A SRC E DST) would be a plausible alternative should the link between B and SRC become over-utilized, if looping could be avoided.

As an alternative to constructing paths so that loops are impossible, a modification to the routing procedure can be adopted which will guarantee loop free paths. The modified routing procedure assumes that each data packet contains a field which can indicate a path's length, in hops; the field should be wide enough to represent the diameter of the network. The value of this field will be referred to as HI_DIST. When a packet initially enters the network, HI_DIST will be set to the network's diameter plus two.

Using the following procedure, if each node has an accurate view of the network's configuration, then loop free paths are guaranteed. A sketch of a proof for this assertion is given in Appendix B. In the following procedure, "length" means hop count, rather than estimated delay. The decision as to which path to select for a given destination is made as follows:

Appendix A: Guided Adaptive Routing for the DDN

```
If no secondary path exists
then
    choose the primary path;
else
    if a packet arrives from the next hop of either path
    then
        choose the other path;
    else
        if (length of primary) != (length of secondary) &&
           (length of secondary) >= HI_DIST
        then
            choose the primary path;
        else
            if (primary's queue + estimated delay) <=
                  (secondary's queue + estimated delay)
            then
                choose primary path;
            else
                choose secondary path;

If secondary path is chosen
then
    set HI_DIST to min(primary hop count, HI_DIST);
```

The effect of the above procedure is that a packet is allowed to take a step "backwards," to avoid long queue delays, but will then be required to take at least two steps "forward," toward its destination. Though the above procedure is much more complicated than a single table look-up, it does not seem prohibitively complex.

A.4 Packet Switching Simulation

A discrete time simulation was used to test the performance of the guided adaptive algorithm against that of the current ARPANET routing algorithm. The simulation models network behavior at a fairly high level of granularity: individual nodes and links with varying capacities are represented. Simulated network events include individual generation of end to end messages, the routing of individual packets, flooding of update packets containing link delay or link utilization measure, and the blocking of host-tandem traffic due to nodal memory resources being exceeded. In

the simulation, routing decisions are made according to separate routing tables maintained for each simulated node. A record of the hop count and total delay is maintained for each simulated packet. The simulation also models the self-organizing behavior of networks, in that the routing tables are constructed by running an SPF algorithm for each node, using connectivity information reported in update messages. However, the simulation does not model end-to-end protocols or message reassembly.

A.4.1 Routing Experiment

Operation of the guided adaptive algorithm and the ARPANET routing algorithm was simulated for a variety of network "configurations" (i.e., topologies), ranging from 19 to 35 nodes in size. In the simulations, information describing a network's configuration is dynamically loaded at run-time.

Most of the configurations used in these simulations are "random" configurations, generated as follows: The number of nodes and a target average outdegree are given. Note that for N nodes, there are N(N-1)/2 possible distinct edges. In the random configurations, each edge is, *a priori*, equally likely to occur in the random configuration; the probability of an edge being "actualized" is derived from the target average outdegree. The configurations are constrained to be connected.

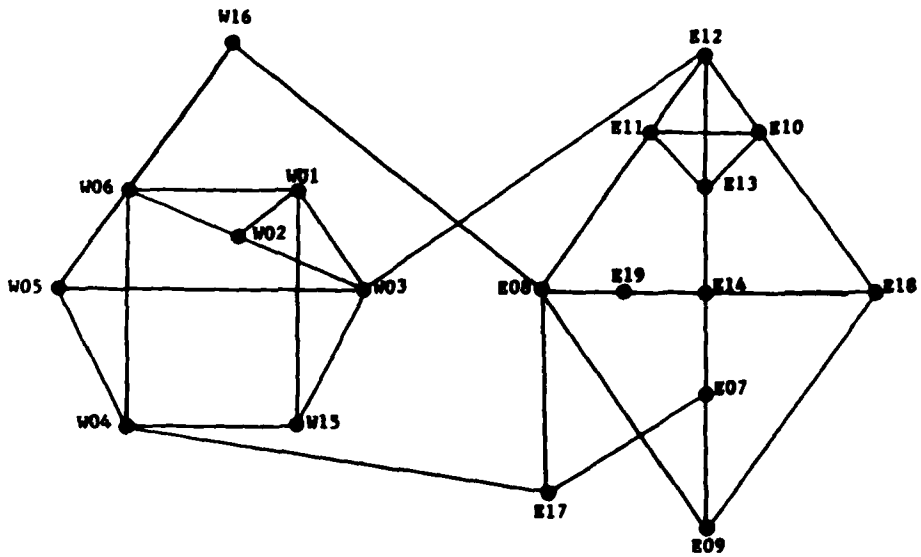
The data traffic of the simulations modeled uniform stochastic message flows between each source and destination pair; various levels of traffic were generated. The traffic events were recorded in "event files," which were read by the simulation program.

By use of the event files and configuration files, the performance of each routing algorithm was compared against the other under identical end-to-end data requirements and network topologies. For both routing

algorithms, the average length of a data packet was set at 750 bits, while the average length of update packets was set to 176 bits. Also, each link in the network was assigned a bandwidth of 56kbs. Finally, in the simulation runs, each routing algorithm was allowed to organize the network prior to the introduction of data traffic.

The performance of the guided adaptive and the ARPANET routing algorithms were compared in terms of the number of data packets which successfully arrived at their destinations within the simulated interval, and the average path length and delay for each of these data packets. In addition, the volume of update traffic, in terms of completed link traversals of update packets, was also recorded. The results of the experiments are summarized in the tables below; the tables describing algorithm performance for a particular network configuration follow an illustration of that configuration. In addition to the illustration, several statistics describing each configuration is given. These include the average outdegree for each node in the configuration, the variance of this value between nodes, and the maximal outdegree for any node. Also, the average min-hop distance for all node pairs is given, followed by the variance over all source-destination pairs, and the "net diameter," or maximal minhop distance. Finally, the percentage of source/destination pairs for which the guided adaptive algorithm cannot generate alternate paths is given; recall that not every destination will have an alternate path

For a given configuration, an event file is identified by its run number. For example, Run 1 for configuration N3101 is identical in terms of end-to-end data traffic loading as Run 1 of configuration N3102. The runs are listed in order of increasing traffic rates.



Configuration N1901
19 Nodes, 34 links
Identical to early ARPANET topology

avg outdegree: 3.58 variance: 1Ø.95 maximal outdegree: 5
avg minhop path: 2.44 variance: 59.45 net diameter: 4
Percentage of guided adaptive routes without alternates: 28.9

Run 1, Configuration N19Ø1

duration: 19.8 seconds offered data traffic: 7198 packets

routing algorithm	data packet arrivals	average path	average delay(msec)	control traffic
guided adaptive	7151	2.51	43.Ø7	248
ARPANET	7134	2.98	93.53	1564

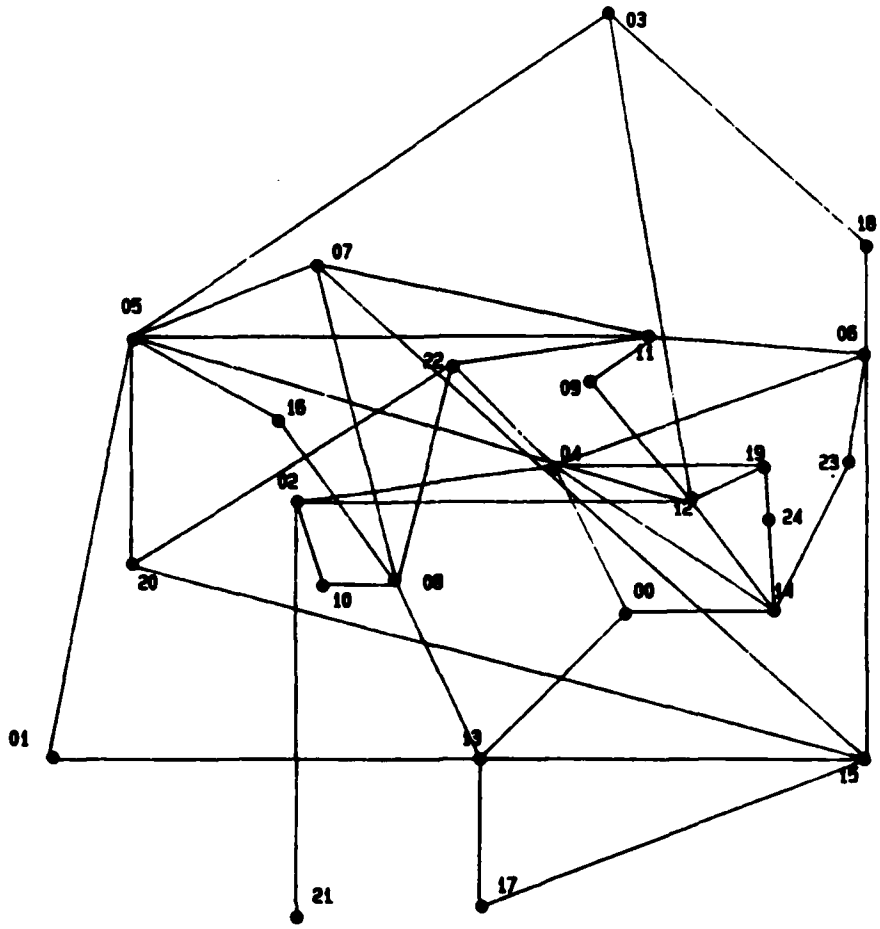
Run 2, Configuration N19Ø1

duration: 23.16 seconds offered data traffic: 16286 packets

routing algorithm	data packet arrivals	average path	average delay(msec)	control traffic
guided adaptive	162Ø8	2.68	581.46	426Ø
ARPANET	55Ø1	4.23*	55.62*	4Ø17

*Note that the ARPANET algorithm has collapsed at this time. Its reported averages for delay and distance are underestimated, in that they are based upon reports from packets which have reached their destinations. However, a significant number of packets remain trapped in the network.

Appendix A: Guided Adaptive Routing for the DDN



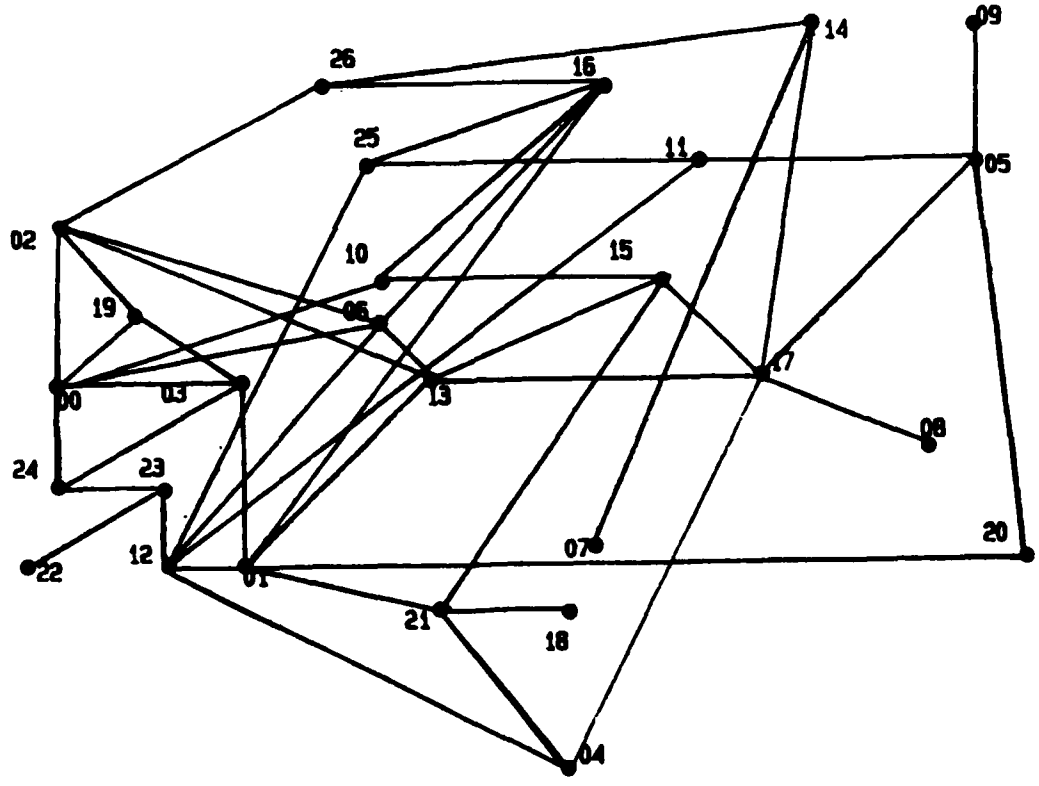
Configuration N25Ø1
25 Nodes, 46 links
Randomly Generated Network

avg outdegree: 3.68 variance: 13.14 maximal outdegree: 7
avg minhop path: 2.42 variance: 76.54 net diameter: 5
Percentage of guided adaptive routes without alternates: 2Ø.8

Run 1, Configuration N25Ø1

duration: 7.8 seconds offered data traffic: 4823 packets

routing algorithm	data packet arrivals	average path	average delay(msec)	control traffic
guided adaptive	4779	2.52	42.15	178
ARPANET	4777	2.52	53.29	736



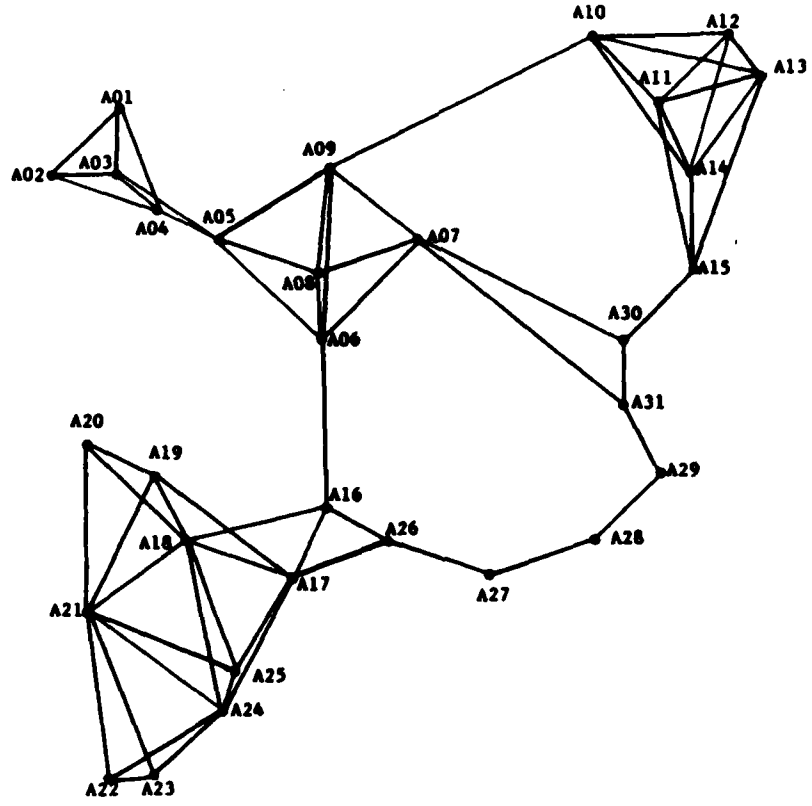
Configuration N2701
27 Nodes, 48 links
Randomly Generated Network

avg outdegree: 3.56 variance: 12.10 maximal outdegree: 6
avg minhop path: 2.65 variance: 101.38 net diameter: 5
Percentage of guided adaptive routes without alternates: 32.5

Run 1, Configuration N2701

duration: 5.8 seconds offered data traffic: 5796 packets

routing algorithm	data packet arrivals	average path	average delay(msec)	control traffic
guided adaptive	5693	2.87	66.02	1407
ARPANET	5407	4.16	265.82	2784



Configuration N31Ø1
31 Nodes, 62 links
Subgraph of 1ØØ node network in [CAL85]

avg outdegree: 4.ØØ variance: 14.13 maximal outdegree: 7
avg minhop path: 3.7Ø variance: 243.68 net diameter: 7
Percentage of guided adaptive routes without alternates: 43.Ø

Run 1, Configuration N31Ø1

duration: 9.8 seconds offered data traffic: 3632 packets

routing algorithm	data packet arrivals	average path	average delay(msec)	control traffic
guided adaptive	3571	3.77	139.99	8Ø6
ARPANET	3583	4.55	139.Ø2	1984

Run 2, Configuration N31Ø1

duration: 7.8 seconds offered data traffic: 4448 packets

routing algorithm	data packet arrivals	average path	average delay(msec)	control traffic
guided adaptive	2465	3.37*	369.24*	3437
ARPANET	361Ø	5.79	484.Ø3	52Ø8

Run 3, Configuration N31Ø1

duration: 6.8 seconds offered data traffic: 4ØØ3 packets

routing algorithm	data packet arrivals	average path	average delay(msec)	control traffic
guided adaptive	2195	3.5Ø*	32Ø.Ø2*	514Ø
ARPANET	3227	5.Ø9	5Ø.65	3968

Run 4, Configuration N31Ø1

duration: 4.8 seconds offered data traffic: 2897 packets

routing algorithm	data packet arrivals	average path	average delay(msec)	control traffic
guided adaptive	2Ø34	3.52*	296.46*	1116
ARPANET	2346	5.14	353.23	3Ø26

*Note that the guided adaptive algorithm has collapsed at this time. Its reported averages for delay and distance are underestimated, in that they are based upon reports from packets which have reached their destinations. However, a significant number of packets remain trapped in the network.

Run 5, Configuration N31Ø1

duration: 9.8 seconds offered data traffic: 6Ø32 packets

routing algorithm	data packet arrivals	average path	average delay(msec)	control traffic
guided adaptive	2371	3.29*	395.85*	22Ø2
ARPANET	3969	5.31	613.Ø9	8332

Run 6, Configuration N31Ø1

duration: 5.8 seconds offered data traffic: 4329 packets

routing algorithm	data packet arrivals	average path	average delay(msec)	control traffic
guided adaptive	2346	3.28*	297.Ø1*	3788
ARPANET	2932	5.Ø2	45Ø.43	4939

Run 7, Configuration N31Ø1

duration: 12.2 seconds offered data traffic: 9838 packets

routing algorithm	data packet arrivals	average path	average delay(msec)	control traffic
guided adaptive	1627	3.32**	331.99**	3698
ARPANET	1612	4.65**	467.87**	4272

*Note that the guided adaptive algorithm has collapsed at this time. Its reported averages for delay and distance are underestimated, in that they are based upon reports from packets which have reached their destinations. However, a significant number of packets remain trapped in the network.

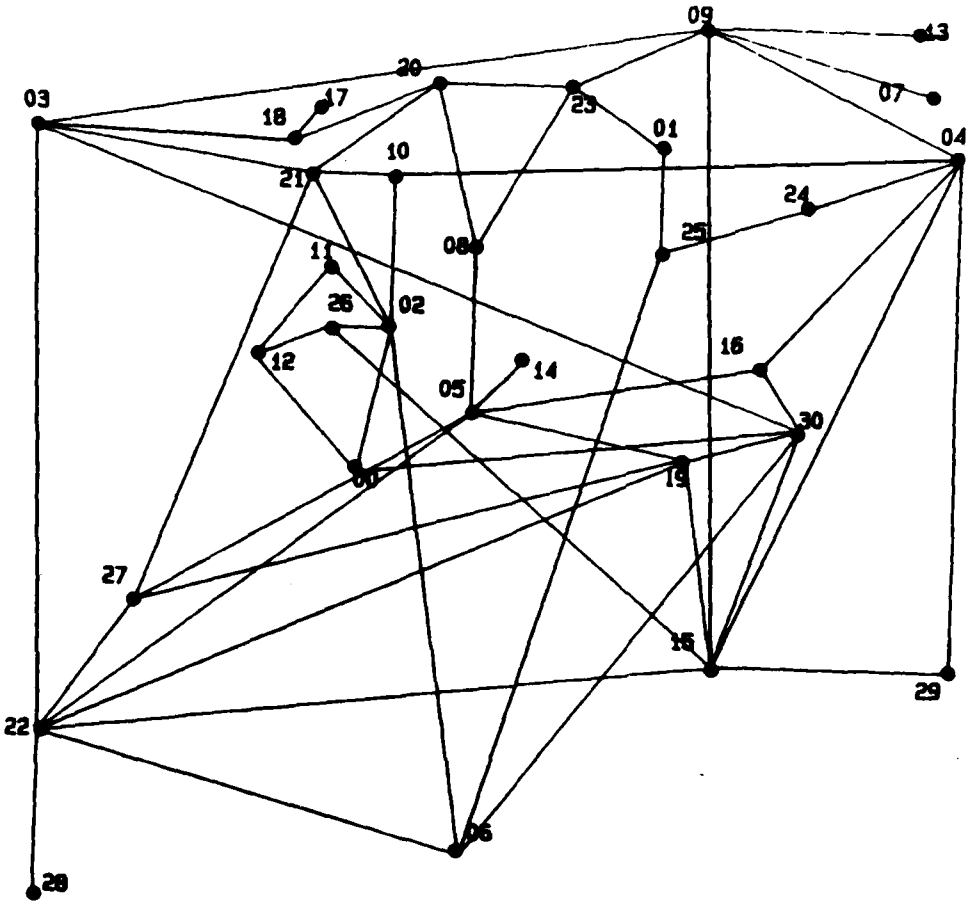
**Both the guided adaptive and flat routing algorithms have collapsed. Neither the average delay nor average hop count are reliable, due to the number of packets which are failing to report as a result of being trapped in the network.

Run 8, Configuration N31Ø1

duration: 5.8 seconds offered data traffic: 5898 packets

routing algorithm	data packet arrivals	average path	average delay(msec)	control traffic
guided adaptive	1369	3.14**	34Ø.29**	1292
ARPANET	989	4.Ø2**	385.29**	3133

**Both the guided adaptive and flat routing algorithms have collapsed. Neither the average delay nor average hop count are reliable, due to the number of packets which are failing to report as a result of being trapped in the network.



Configuration N3102
31 Nodes, 55 links
Randomly Generated Network

avg outdegree: 3.55 variance: 12.95 maximal outdegree: 7
avg minhop path: 2.73 variance: 123.54 net diameter: 5
Percentage of guided adaptive routes without alternates: 32.9

Run 1, Configuration N3102

duration: 9.8 seconds offered data traffic: 3632 packets

routing algorithm	data packet arrivals	average path	average delay(msec)	control traffic
guided adaptive	3616	2.79	43.29	0
ARPANET	3614	2.95	54.01	1210

Run 2, Configuration N31Ø2

duration: 7.8 seconds offered data traffic: 4448 packets

routing algorithm	data packet arrivals	average path	average delay(msec)	control traffic
guided adaptive	4396	2.83	46.78	44Ø
ARPANET	4375	3.ØØ	68.65	132Ø

Run 3, Configuration N31Ø2

duration: 6.8 seconds offered data traffic: 4ØØ3 packets

routing algorithm	data packet arrivals	average path	average delay(msec)	control traffic
guided adaptive	395Ø	2.73	45.54	4Ø4
ARPANET	3927	2.82	57.Ø9	11ØØ

Run 4, Configuration N31Ø2

duration: 4.8 seconds offered data traffic: 2897 packets

routing algorithm	data packet arrivals	average path	average delay(msec)	control traffic
guided adaptive	2853	2.86	5Ø.28	456
ARPANET	2838	2.85	6Ø.73	1Ø76

Run 5, Configuration N31Ø2

duration: 9.8 seconds offered data traffic: 6Ø32 packets

routing algorithm	data packet arrivals	average path	average delay(msec)	control traffic
guided adaptive	598Ø	2.79	47.22	5Ø2
ARPANET	5964	3.23	87.84	198Ø

Run 6, Configuration N31Ø2

duration: 5.8 seconds offered data traffic: 4329 packets

routing algorithm	data packet arrivals	average path	average delay(msec)	control traffic
guided adaptive	426Ø	2.85	51.84	152
ARPANET	4227	3.Ø6	78.23	143Ø

Run 7, Configuration N31Ø2

duration: 12.2 seconds offered data traffic: 9838 packets

routing algorithm	data packet arrivals	average path	average delay(msec)	control traffic
guided adaptive	9793	2.85	52.69	2144
ARPANET	9761	3.4Ø	1Ø4.73	297Ø

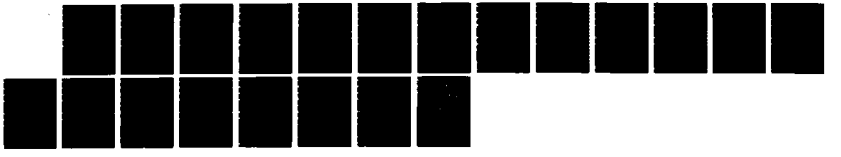
Run 8, Configuration N31Ø2

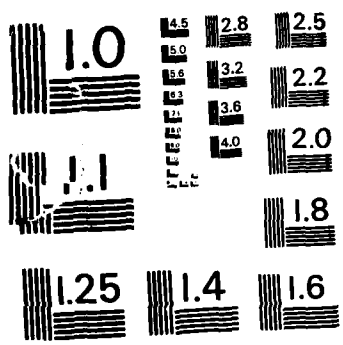
duration: 5.8 seconds offered data traffic: 5898 packets

routing algorithm	data packet arrivals	average path	average delay(msec)	control traffic
guided adaptive	58ØØ	2.84	6Ø.79	8Ø3
ARPANET	5711	3.78	182.64	275Ø

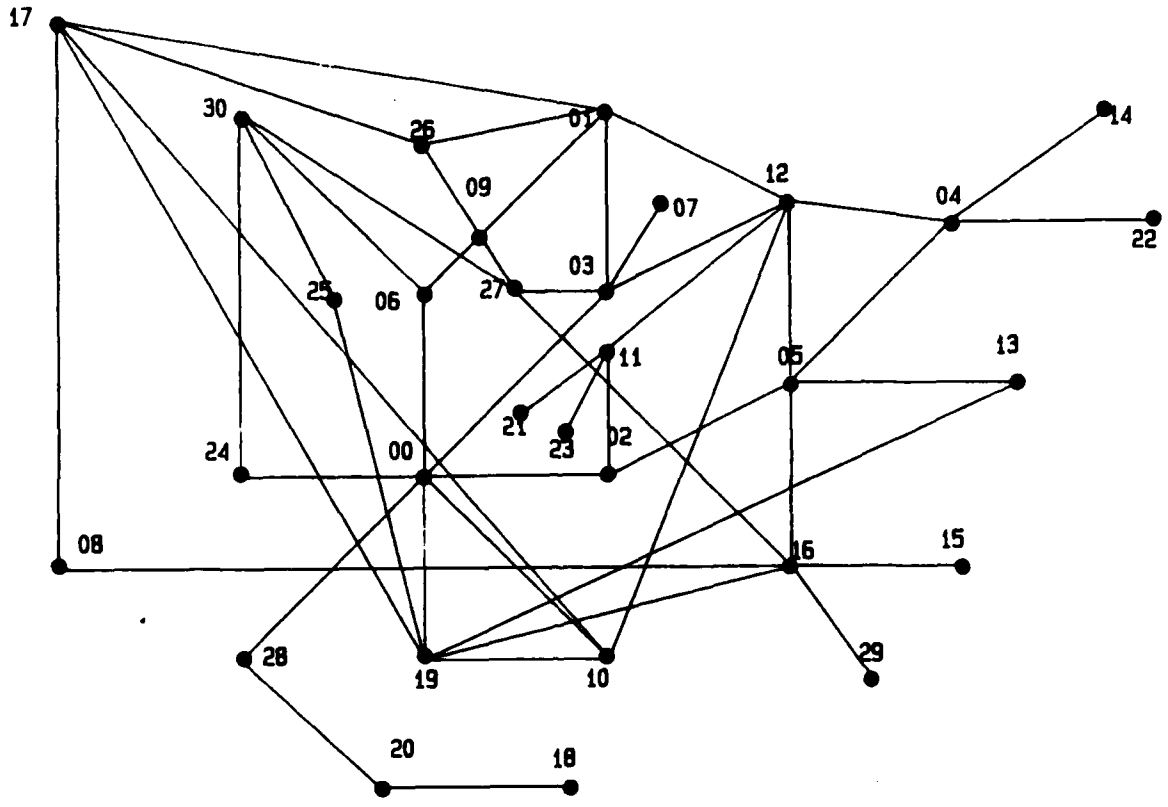
AD-A167 619 DESIGN AND ANALYSIS FOR AREA ROUTING IN LARGE NETWORKS 2/2
(U) SPARTA INC MCLEAN VA 22 APR 86 URW-6-DCA-052
DCA100-84-C-0085

UNCLASSIFIED F/G 9/3 NL





MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS - 1963 - A



Configuration N3103
31 Nodes, 48 links
Randomly Generated Network

avg outdegree: 3.10 variance: 10.59 maximal outdegree: 7
avg minhop path: 3.05 variance: 158.88 net diameter: 7
Percentage of guided adaptive routes without alternates: 47.7

Run 1, Configuration N3103

duration: 9.8 seconds offered data traffic: 3632 packets

routing algorithm	data packet arrivals	average path	average delay(msec)	control traffic
guided adaptive	3632	3.11	51.00	70
ARPANET	3632	3.10	57.27	960

Run 2, Configuration N31Ø3

duration: 7.8 seconds offered data traffic: 4448 packets

routing algorithm	data packet arrivals	average path	average delay(msec)	control traffic
guided adaptive	4391	3.1Ø	62.3Ø	776
ARPANET	4373	3.15	76.73	96Ø

Run 3, Configuration N31Ø3

duration: 6.8 seconds offered data traffic: 4ØØ3 packets

routing algorithm	data packet arrivals	average path	average delay(msec)	control traffic
guided adaptive	3939	3.16	66.92	4Ø4
ARPANET	3924	3.85	57.Ø9	1632

Run 4, Configuration N31Ø3

duration: 4.8 seconds offered data traffic: 2897 packets

routing algorithm	data packet arrivals	average path	average delay(msec)	control traffic
guided adaptive	2842	3.17	65.13	389
ARPANET	2897	3.38	95.Ø1	1138

Run 5, Configuration N31Ø3

duration: 9.8 seconds offered data traffic: 6Ø32 packets

routing algorithm	data packet arrivals	average path	average delay(msec)	control traffic
guided adaptive	5979	3.14	79.21	2666
ARPANET	5967	3.51	113.23	1632

Run 6, Configuration N31Ø3

duration: 5.8 seconds offered data traffic: 4329 packets

routing algorithm	data packet arrivals	average path	average delay(msec)	control traffic
guided adaptive	4255	3.Ø7	69.28	797
ARPANET	42Ø4	3.73	142.95	192Ø

Run 7, Configuration N31Ø3

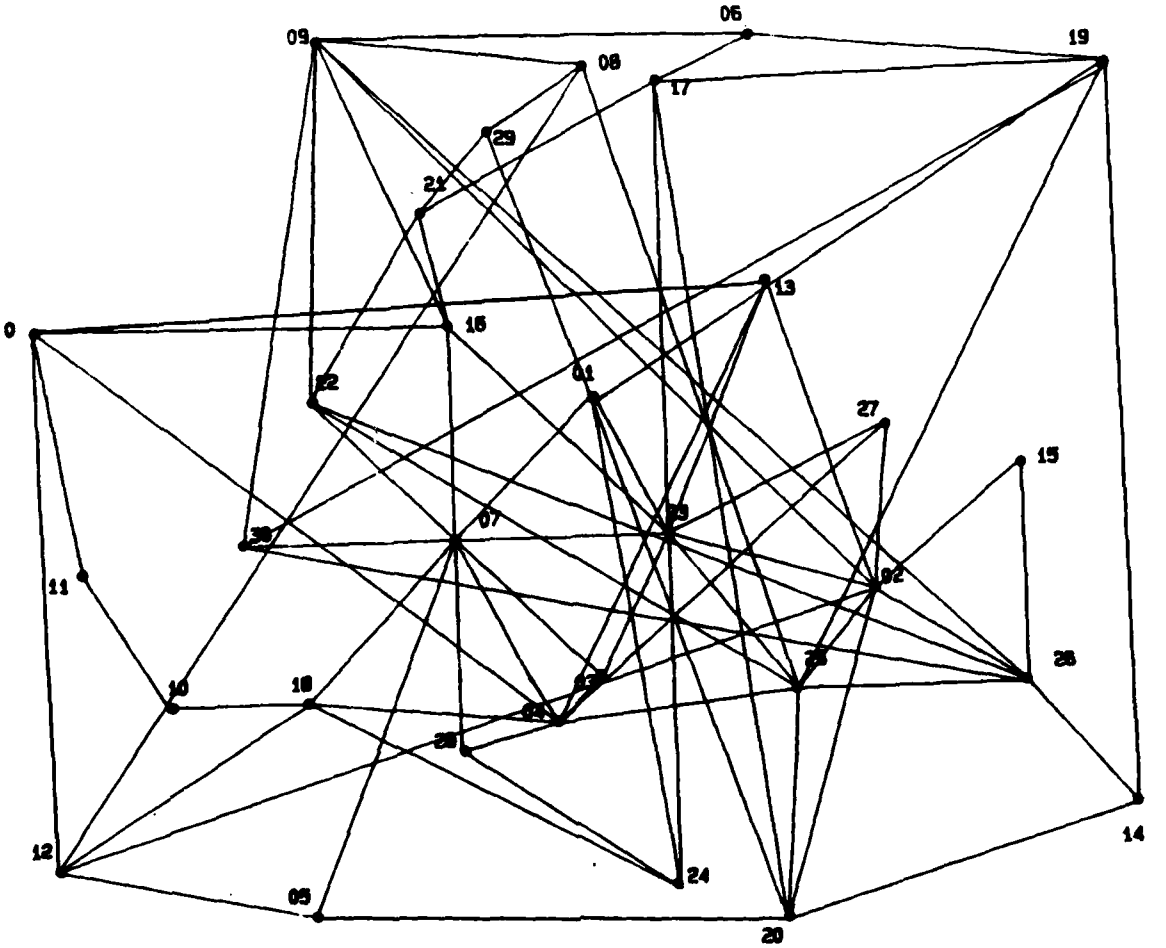
duration: 12.2 seconds offered data traffic: 9838 packets

routing algorithm	data packet arrivals	average path	average delay(msec)	control traffic
guided adaptive	9792	3.11	1ØØ.36	4141
ARPANET	9734	4.1Ø	229.Ø7	47Ø6

Run 8, Configuration N31Ø3

duration: 5.8 seconds offered data traffic: 5898 packets

routing algorithm	data packet arrivals	average path	average delay(msec)	control traffic
guided adaptive	4778	3.Ø4	168.65	1612
ARPANET	4685	4.23	379.22	3Ø51



Configuration N31Ø4
31 Nodes, 77 links
Randomly Generated Network

avg outdegree: 4.97 variance: 24.29 maximal outdegree: 1Ø
avg minhop path: 2.27 variance: 85.29 net diameter: 5
Percentage of guided adaptive routes without alternates: 17.6

Run 1, Configuration N31Ø4

duration: 9.8 seconds offered data traffic: 3632 packets

routing algorithm	data packet arrivals	average path	average delay(msec)	control traffic
guided adaptive	3617	2.35	34.74	16
ARPANET	3617	2.32	38.73	154Ø

Run 2, Configuration N31Ø4

duration: 7.8 seconds offered data traffic: 4448 packets

routing algorithm	data packet arrivals	average path	average delay(msec)	control traffic
guided adaptive	44ØØ	2.37	36.28	7Ø
ARPANET	4399	2.34	42.42	1386

Run 3, Configuration N31Ø4

duration: 6.8 seconds offered data traffic: 4ØØ3 packets

routing algorithm	data packet arrivals	average path	average delay(msec)	control traffic
guided adaptive	3965	2.33	35.53	132
ARPANET	3956	2.41	5Ø.65	1386

Run 4, Configuration N31Ø4

duration: 4.8 seconds offered data traffic: 2897 packets

routing algorithm	data packet arrivals	average path	average delay(msec)	control traffic
guided adaptive	2865	2.32	36.54	176
ARPANET	2897	2.28	42.61	1Ø76

Run 5, Configuration N31Ø4

duration: 9.8 seconds offered data traffic: 6Ø32 packets

routing algorithm	data packet arrivals	average path	average delay(msec)	control traffic
guided adaptive	599Ø	2.35	36.5Ø	1476
ARPANET	5988	2.42	47.64	1694

Run 6, Configuration N31Ø4

duration: 5.8 seconds offered data traffic: 4329 packets

routing algorithm	data packet arrivals	average path	average delay(msec)	control traffic
guided adaptive	4273	2.36	37.21	738
ARPANET	4268	2.48	53.1Ø	1386

Run 7, Configuration N31Ø4

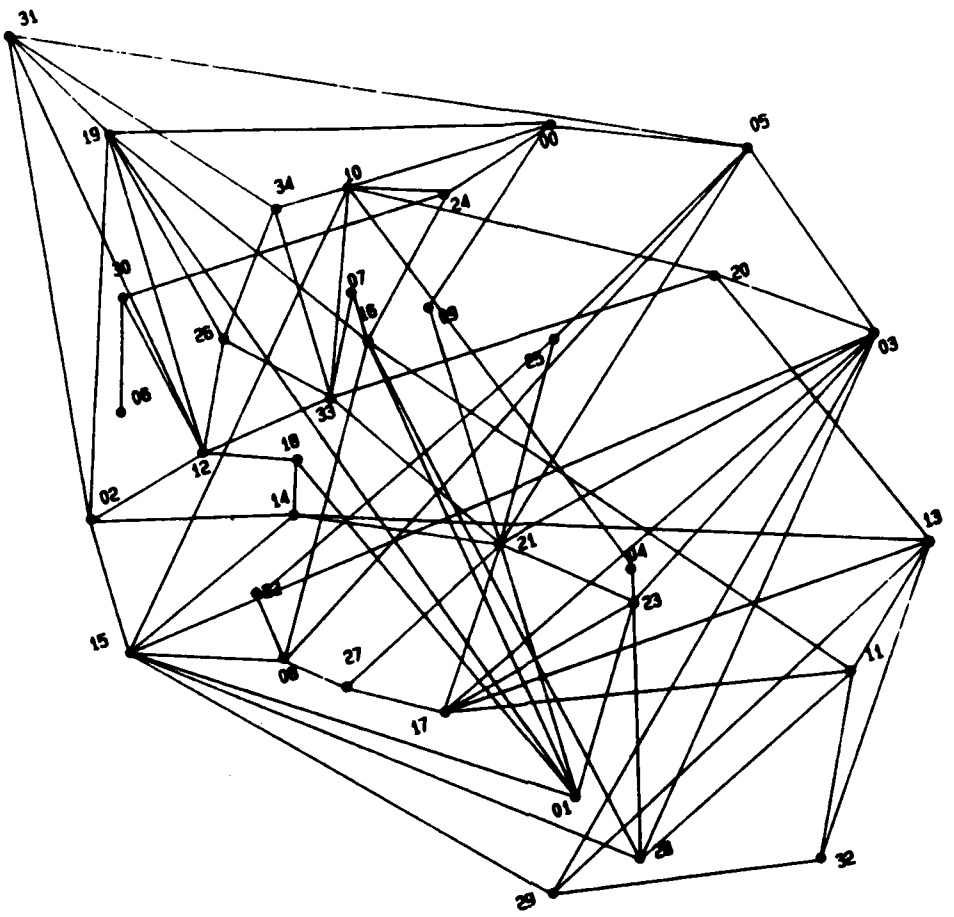
duration: 12.2 seconds offered data traffic: 9838 packets

routing algorithm	data packet arrivals	average path	average delay(msec)	control traffic
guided adaptive	98Ø8	2.38	37.9Ø	7ØØ
ARPANET	98Ø4	2.55	56.62	2464

Run 8, Configuration N31Ø4

duration: 5.8 seconds offered data traffic: 5898 packets

routing algorithm	data packet arrivals	average path	average delay(msec)	control traffic
guided adaptive	5817	2.39	4Ø.18	1522
ARPANET	5762	2.52	59.7Ø	1694



Configuration N35Ø1
35 Nodes, 85 links
Randomly Generated Network

avg outdegree: 4.86 variance: 22.46 maximal outdegree: 1Ø
avg minhop path: 2.4Ø variance: 1Ø8.Ø7 net diameter: 5
Percentage of guided adaptive routes without alternates: 18.Ø

Run 1, Configuration N35Ø1

duration: 5.8 seconds offered data traffic: 2Ø38 packets

routing algorithm	data packet arrivals	average path	average delay(msec)	control traffic
guided adaptive	2Ø18	2.4Ø	34.74	28
ARPANET	2Ø38	2.34	38.46	17ØØ

Run 2, Configuration N35Ø1

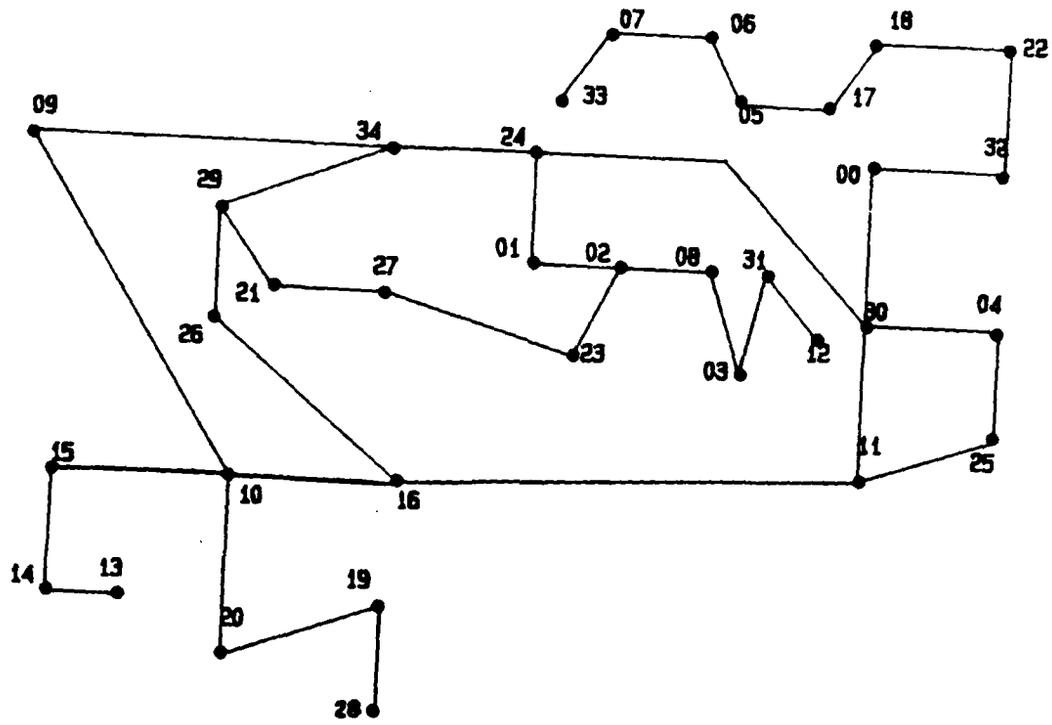
duration: 4.8 seconds offered data traffic: 9Ø85 packets

routing algorithm	data packet arrivals	average path	average delay(msec)	control traffic
guided adaptive	8925	2.55	52.43	6664
ARPANET	7416	4.22	326.83	7824

Run 3, Configuration N35Ø1

duration: 5.Ø seconds offered data traffic: 1Ø229 packets

routing algorithm	data packet arrivals	average path	average delay(msec)	control traffic
guided adaptive	1ØØ5Ø	2.55	55.1Ø	8822
ARPANET	5784	4.2Ø	3Ø9.79	6977



Configuration N35Ø2
35 Nodes, 38 links
Randomly Generated Network

avg outdegree: 2.17 variance: 4.14 maximal outdegree: 4
avg minhop path: 6.Ø4 variance: 799.Ø9 net diameter: 1Ø
Percentage of guided adaptive routes without alternates: 76.8

Run 1, Configuration N35Ø2

duration: 5.8 seconds

offered data traffic: 2Ø38 packets

routing algorithm	data packet arrivals	average path	average delay(msec)	control traffic
guided adaptive	1993	6.2Ø	145.2Ø	312
ARPANET	1972	6.32	167.67	988

Run 2, Configuration N35Ø2

duration: 4.8 seconds offered data traffic: 9Ø85 packets

routing algorithm	data packet arrivals	average path	average delay(msec)	control traffic
guided adaptive	931	3.98**	43Ø.19**	7Ø8
ARPANET	791	4.Ø6**	394.97**	1376

Run 3, Configuration N35Ø2

duration: 5.Ø seconds offered data traffic: 1Ø229 packets

routing algorithm	data packet arrivals	average path	average delay(msec)	control traffic
guided adaptive	82Ø	3.9Ø**	413.8Ø**	522
ARPANET	632	3.87**	353.78**	1378

**Both the guided adaptive and flat routing algorithms have collapsed. Neither the average delay nor average hop count are reliable, due to the number of packets which are failing to report as a result of being trapped in the network.

A.4.2 Interpretation of Results

For all combinations of configuration and traffic loads except for a moderately or highly loaded N31Ø1, the guided adaptive algorithm had markedly lower average packet delay than the ARPANET algorithm. In addition, in most cases the total throughput of the guided adaptive algorithm was superior to the ARPANET algorithm. These effects are probably due to the high degree of load splitting which the guided adaptive algorithm performs. The poor performance of the of the guided adaptive algorithm for N31Ø1 seems to be due to the fact that it consists of several highly connected clusters which are linked by relatively few paths, a topological feature that seems somewhat correlated with a high variance in minhop path distances. Such a configuration would be relatively easy to partition. Since the DDN is designed so as to be difficult to partition, it is likely that the configurations for which the guided adaptive algorithm performs poorly might not arise in that network.

No weakening is seen in the relative performance of the guided adaptive algorithm as the size of configurations increase. This suggests that the algorithm could "scale up" for large networks. However, because of the small number of nodes in these experiments, no firm conclusion can be reached on this issue.

The guided adaptive algorithm produced path lengths of slightly fewer hops than the ARPANET algorithm. This is not too surprising, since the ARPANET SPF algorithm weights hops according to average delay. Hence, the ARPANET algorithm will often reroute traffic so that it makes more hops, over links of lower total delay.

In the experiment, the volume of control traffic generated by the guided adaptive algorithm tended to be less than that generated by the

ARPANET algorithm. However, only limited significance should be attached to this observation, since the ARPANET algorithm transmitted updates at average intervals ranging from 7 to 3Ø seconds. This rate is somewhat higher than the observed average of 3Ø seconds reported for the actual ARPANET. Yet this finding does not contradict the more general finding of the competitive traffic handling capabilities of the guided adaptive algorithm, inasmuch as in some cases the the guided adaptive algorithm out performed the ARPANET algorithm even when the guided adaptive algorithm generated the greater volume of control traffic (e.g., see Run 5 of configuration N31Ø3, and Run 3, N35Ø1).

A.5 Areas for Further Research

As the simulation results show, the guided adaptive algorithm shows some promise as a high performance routing algorithm for large networks. Before a strong recommendation can be made for the guided adaptive approach, several topics should be investigated. Most important among these would be to extend the simulations to large networks, since potentially disastrous effects might not appear in a 35 node network. Related to this topic would be a study into the relative frequency of updates between the guided adaptive and the ARPANET approaches. If the guided adaptive approach required a significantly higher volume of update traffic, then its performance could be inferior to that of the current ARPANET algorithm.

Related to the issue of update frequency is the question of optimal queue threshold values. A more fundamental question is whether queue lengths are stable enough to provide good estimates of delays to the guided adaptive algorithm. Yet even if queue lengths proved to be unsatisfactory for estimating delays, the guided adaptive approach could still be

implemented, by basing path delay estimates on significant changes in round trip link delay. In that case, a guided adaptive scheme could certainly be developed which requires updates at no greater frequency than the current ARPANET. Given an update frequency roughly equal to that of the ARPANET, the simulation results suggest that the multipath nature of the guided adaptive scheme should provide lower average delay.

Yet another topic for further study would be to bound more closely the type of network configurations for which the guided adaptive algorithm fails. Such a study would be expected to shed light on the operation of the guided adaptive algorithm, as well as on its suitability for the DDN or other networks.

There are several design changes that could be beneficial for the guided adaptive routing algorithm. Most important would be a means to relax the constraint that alternate paths must be no more than two hops longer than primary paths; this would reduce the number of source/destination pairs which lack alternate paths. However, this would require the establishment of a new loop-free routing procedures, which is a non-trivial task (see Appendix B).

Another major design that could improve the performance of the guided adaptive algorithm would be to modify the selection procedure for primary vs. secondary paths. Since the good performance of the guided adaptive approach seems to be due to load balancing, a mechanism for path selection which is more stable and fair than the currently used local queue sizes might be used.

A major design issue which must be addressed before the guided adaptive routing algorithm could be fielded is the development of a procedure for bringing nodes into a running network. The major issue that

must be resolved is the development of a means for determining the existing utilization levels of the network's links. The two major alternatives are: 1) have each original node of the network send link utilization information to the newly arrived node; and 2) make the initial assumption that all links are at low levels of utilization, but alter this assumption as update packets are received that would indicate otherwise.

Less radical improvements might also be made to the guided adaptive algorithm. For example, to account for the large difference in average lengths of update and data packets - the average ARPANET update packet is about 200 bits long, while the average data packet is about 700 bits long - a "weighted" queue length instead of a simple packet count might be maintained. In fact, in the simulation of the guided adaptive technique, only data packets in a queue were used to estimate local queuing delay. Another enhancement that would bear investigation would be including information on all of a node's incident links in an update message. This would allow a more current view of the network to be propagated, at relatively little additional cost.

In summary, the guided adaptive approach to routing appears promising, and also raises interesting questions for further study.

Appendix B

Proof that Guided Adaptive Algorithm is Loop-free

The following notation will be used:

$pri(N_\emptyset,N_1)$	Primary path from node $N_\emptyset$ to node N_1.
$sec(N_\emptyset,N_1)$	Secondary path from $N_\emptyset$ to N_1.
Next(q)	First hop in path q.
L(q)	Length of path q, in hops.
$d(N_\emptyset,DST)$	Minimum hop distance from node $N_\emptyset$ to node DST.
HI_DIST	A field in the packet, initially set to network diameter plus two, but decreased to $min(HI\_DIST,d(N_\emptyset,DST))$ if a node $N_\emptyset$ uses a secondary path to forward the packet.

The network is assumed to be connected, and all nodes are assumed to have consistent views of the network's configuration. In addition, from the procedure of generating secondary paths, it is known that $Next(pri(N_\emptyset,N_n))$ is distinct from $Next(sec(N_\emptyset,N_n))$. Furthermore, $pri(N_\emptyset,N_n)$ is a minimal length path, which is to say that $L(pri(N_\emptyset,N_n)) = d(N_\emptyset,N_n)$. Also, $L(pri(N_\emptyset,N_n)) <= L(sec(N_\emptyset,N_n))$, but by the constraints on the formation of secondary paths, $L(sec(N_\emptyset,N_n)) <= L(pri(N_\emptyset,N_n)) +2$.

There is no guarantee that $sec(N_\emptyset,N_n)$ exists. However, if a packet arrives at N_1 from $N_\emptyset$, there must be a path from N_1 to the destination which does not include $N_\emptyset$, as both primary and secondary paths are constructed by use of the SPF algorithm, and do not include cycles. Finally, it is assumed that the nodes will follow sufficiently uniform procedures for computing their primary paths, so that if the primary path from a node $N_\emptyset$ to a given destination passes through node N_1, then from N_1 on, the primary path of N_1 to the destination will coincide with the primary path from $N_\emptyset$.

Appendix B: Proof of Loop-free Algorithm

In the discussion below, the "no rebound" constraint refers to the routing algorithm prohibition against transmitting a packet immediately back to the node from which it was received. The HI_DIST constraint refers to a limitation which the routing algorithm imposes on choosing a secondary path in preference to a primary path, which, for a node N, could be express as the rule:

If L(sec(N, DST)) != L(pri(N,DST)) and
    L(sec(N,DST)) >= HI_DIST,
then
    the primary path must be selected.

Stating the rule as its contrapositive, if a secondary path is selected, then either L(sec(N,DST)) < HI_DIST, or L(sec(N,DST)) = L(pri(N,DST)).

Since in moving one hop, the distance can be changed at most by 1, if a packet with destination DST arrives at N_1 from node $N_\emptyset$, then three cases are possible:

I. $d(N_1,DST) = d(N_\emptyset,DST) + 1$, which would occur if the primary path from N_1 to DST is either through $N_\emptyset$ or through another node which is $d(N_\emptyset,DST)$ from DST; the packet would have taken a step backwards;

II. $d(N_1,DST) = d(N_\emptyset,DST)$, which would occur if $N_\emptyset$ and N_1 have different, though equally long, primary paths to DST; the packet would have made no progress but lost no ground in its journey to DST;

III. $d(N_1,DST) = d(N_\emptyset,DST) - 1$, which would occur if the primary path from $N_\emptyset$ to DST is either through N_1 or through another node which is $d(N_1,DST)$ from DST; the packet would have taken a step towards DST.

In both case I and II, it will be shown that upon reaching node N_3, the third intermediate node in the journey from $N_\emptyset$ to DST, a packet will be

at most $d(N_\emptyset,DST) - 1$ from DST. In case III, a packet at N_1 is already arrived at distance $d(N_\emptyset,DST) - 1$ of DST. For a packet leaving N_1, it will be shown that either the reasoning from case I or case II can be applied, or it will be trivially true that the packet has approached DST even closer. Hence, if $d(N_\emptyset,DST)$ is 2 or greater, then a packet that is $d(N_\emptyset,DST) - 1$ from DST is ensured of reaching a node which is no more than $d(N_\emptyset,DST) - 2$ from DST.

Since these proofs apply to any arbitrary node along the packet's route - any node along the path can be considered $N_\emptyset$ - and since $d(N_\emptyset,DST)$, the minimum hop distance from $N_\emptyset$ to DST, is finite, then the path navigated by a packet must be finite, and is bounded above by $3(d(N_\emptyset,DST))$. Furthermore, the rule that a packet must take a primary path if no secondary path exists insures that a packet will reach DST.

case I. $d(N_1,DST) = d(N_\emptyset,DST) + 1$.

The route taken by the packet is an alternate route. Hence, HI_DIST $<= d(N_\emptyset,DST)$. The path $pri(N_1,DST)$ must either have $N_\emptyset$ as a next hop, or not.

subcase I.i. $d(N_1,DST) = d(N_\emptyset,DST) + 1$, and $Next(pri(N_1,DST)) != N_\emptyset$.

Because of the HI_DIST and "no rebound" constraints of the routing algorithm, either the primary path must be selected, or both $L(sec(N_1,DST)) = L(pri(N_1,DST))$ and $Next(sec(N_1,DST)) != N_\emptyset$. Since, by the assumption of case I, $d(N_1,DST) = d(N_\emptyset,DST) + 1$, then any allowable next hop, N_2, is at most $d(N_\emptyset,DST)$ from DST (i.e., it is one step closer than to DST than N_1).

Node N_2 must now apply the routing algorithm to the packet which has traveled from $N_\emptyset$ through N_1, to N_2. We have established that $d(N_2,DST) = d(N_\emptyset,DST)$, and HI_DIST $<= d(N_\emptyset,DST)$. Now, $Next(pri(N_2,DST)) != N_1$, since

$d(N_1,DST) > d(N_2,DST)$. Hence, the rebound constraint would not be in effect for the primary path from N_2 to DST. Also, the HI_DIST constraint would prohibit selection of the secondary path unless $L(sec(N_2,DST)) = L(pri(N_2,DST))$. Therefore, the next hop reached by the packet, N_3, must be on the primary path or one equivalently short. Hence, $d(N_3,DST) = d(N_2,DST) - 1$, so $d(N_3,DST) = d(N_\emptyset,DST) - 1$.

subcase I.ii. $d(N_1,DST) = d(N_\emptyset,DST) + 1$, and $Next(pri(N_1,DST)) = N_\emptyset$.
By the "no rebound" constraint, N_1's secondary path must be chosen. By the procedure for forming secondary paths, $Next(sec(N_1,DST)) != N_\emptyset$. But since $N_\emptyset$ chose its secondary path to DST in forwarding the packet to N_1, then by definition, $Next(sec(N_\emptyset,DST)) = N_1$. By the constraint on the length of secondary paths, $L(sec(N_\emptyset,DST)) <= d(N_\emptyset,DST) + 2$. Furthermore, since $sec(N_\emptyset,DST)$ has no cycles, and in following $sec(N_\emptyset,DST)$ a packet will be one hop further along the path when N_1 is reached, there is a path from N_1 to DST which does not contain $N_\emptyset$, whose length is $<= d(N_\emptyset,DST) + 1$. Since the secondary path is a minimal length path which does not contain the next hop of the primary path, then $L(sec(N_1,DST)) <= d(N_\emptyset,DST) + 1$.

Let N_2 represent $Next(sec(N_1,DST))$. Since $L(sec(N_1,DST)) <= d(N_\emptyset,DST) + 1$, then $d(N_2,DST) <= d(N_\emptyset,DST)$. Also, N_2 is distinct from $N_\emptyset$, since by subcase I.ii $N_\emptyset$ is the next hop of the primary path from N_1 to DST. Let N_3 be the next hop of the path chosen by N_2. Since the next hop along the primary path is $<= d(N_\emptyset,DST) - 1$ from DST, then by the HI_DIST constraint, $d(N_3,DST) <= d(N_\emptyset,DST) - 1$.

Since subcase i and ii exhaust case I, we have established that a packet which takes a secondary path to a node further from the destination than its previous node will progress to a node closer to the destination within two or fewer subsequent hops. More clearly, if a packet takes a step back, then it must take two forward.

case II. $d(N_1,DST) = d(N_\emptyset,DST)$.

The route taken by the packet is an alternate route. Hence, HI_DIST <= $d(N_\emptyset,DST)$. The path $pri(N_1,DST)$ cannot have $N_\emptyset$ as a next hop, since that would imply that $d(N_\emptyset,DST) = d(N_\emptyset,DST) - 1$.

By the HI_DIST constraint, then if the secondary route is chosen, then $L(sec(N_1,DST)) = d(N_1,DST)$. Hence, which ever path is chosen, if N_2 is the next hop, $d(N_2,DST) = d(N_1,DST) - 1 = d(N_\emptyset,DST) - 1$. So, if the choice of a secondary path sends a packet to a node that is no closer to its destination than its previous packet, then the following node it reaches must be one hop closer.

case III. $d(N_1,DST) = d(N_\emptyset,DST) - 1$.

Trivially, the packet is closer to its destination. Furthermore, if N_1 is not already the destination, then the packet is guaranteed to get even closer to DST. Consider the node N_2, which receives the packet from N_1. The distance, $d(N_2,DST)$, from N_2 to DST, is either one greater, the same, or one less than $d(N_1,DST\emptyset)$. If one greater or the same, then by cases I and II above, the packet is guaranteed to be routed to a node which is no more than $d(N_1,DST) - 1$ from DST. In the remaining case, the packet has been sent on an optimal route toward DST, and so again is no more than $d(N_1,DST) - 1$ from DST.

END

DTIC

6-86